Original decisiontrees Slides

Decision Trees

CS229: Machine Learning
Carlos Guestrin
Stanford University

Slides include content developed by and co-developed with Emily Fox

Predicting potential loan defaults

What makes a loan risky?

I want a to buy
a new house!

Loan
Application

Credit History
★★★★

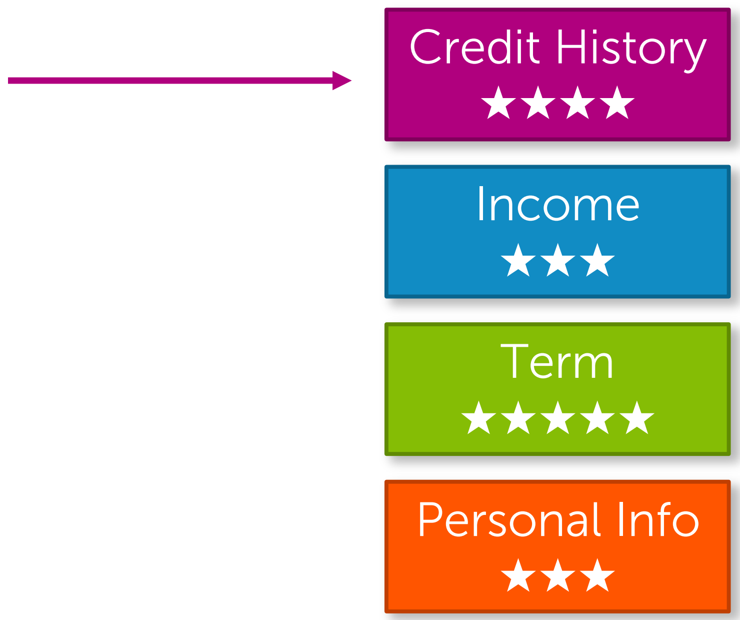
Income
★★★

Term
★★★★★

Personal Info
★★★

Credit history explained

Did I pay previous loans on time?

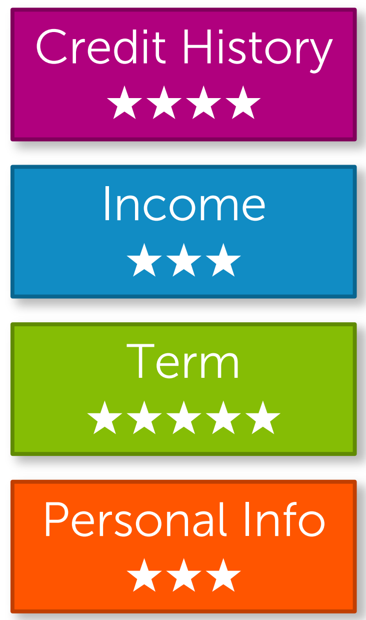


Example:
excellent, good, or fair

Income

What's my income?

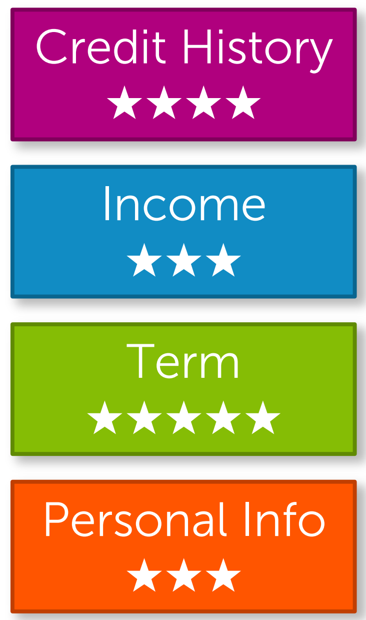
Example:
$80K per year



Loan terms

How soon do I need to pay the loan?

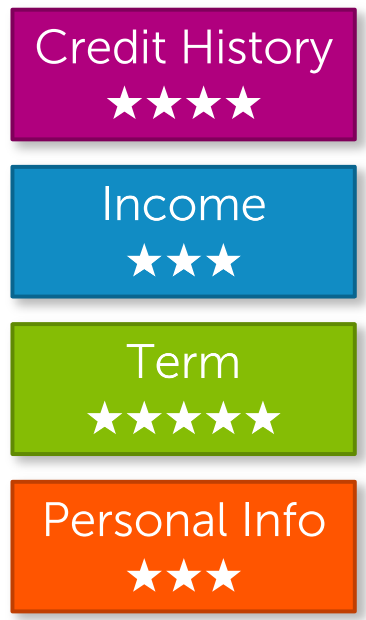
Example: 3 years, 5 years,…



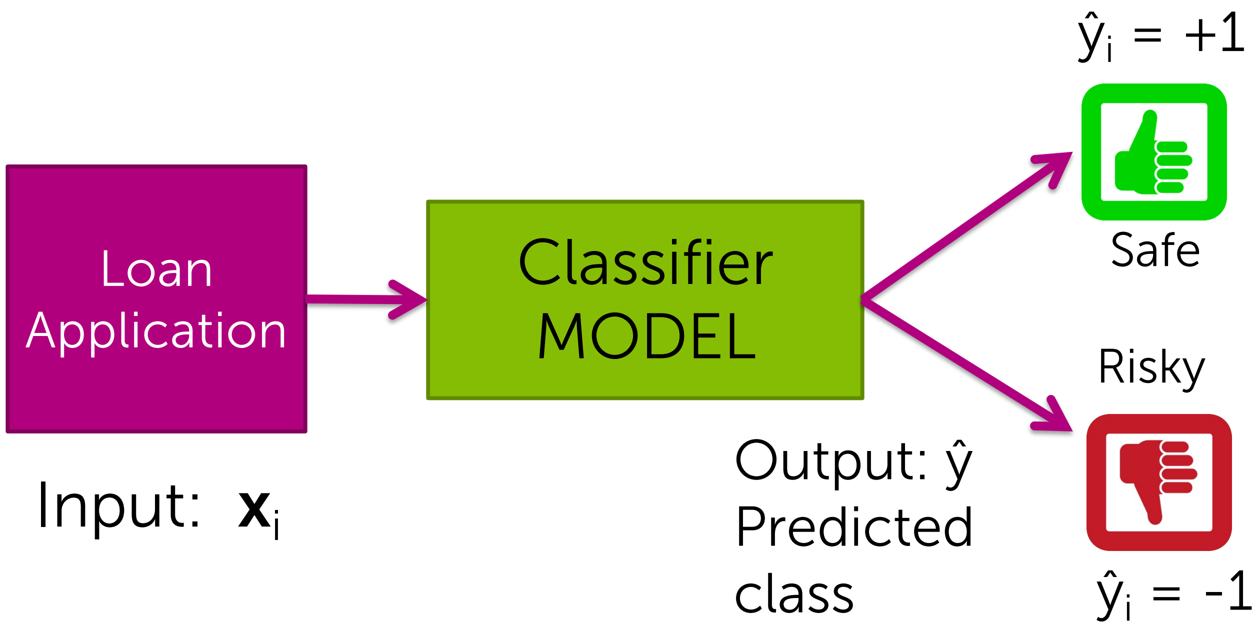
Personal information

Age, reason for the loan, marital status,…

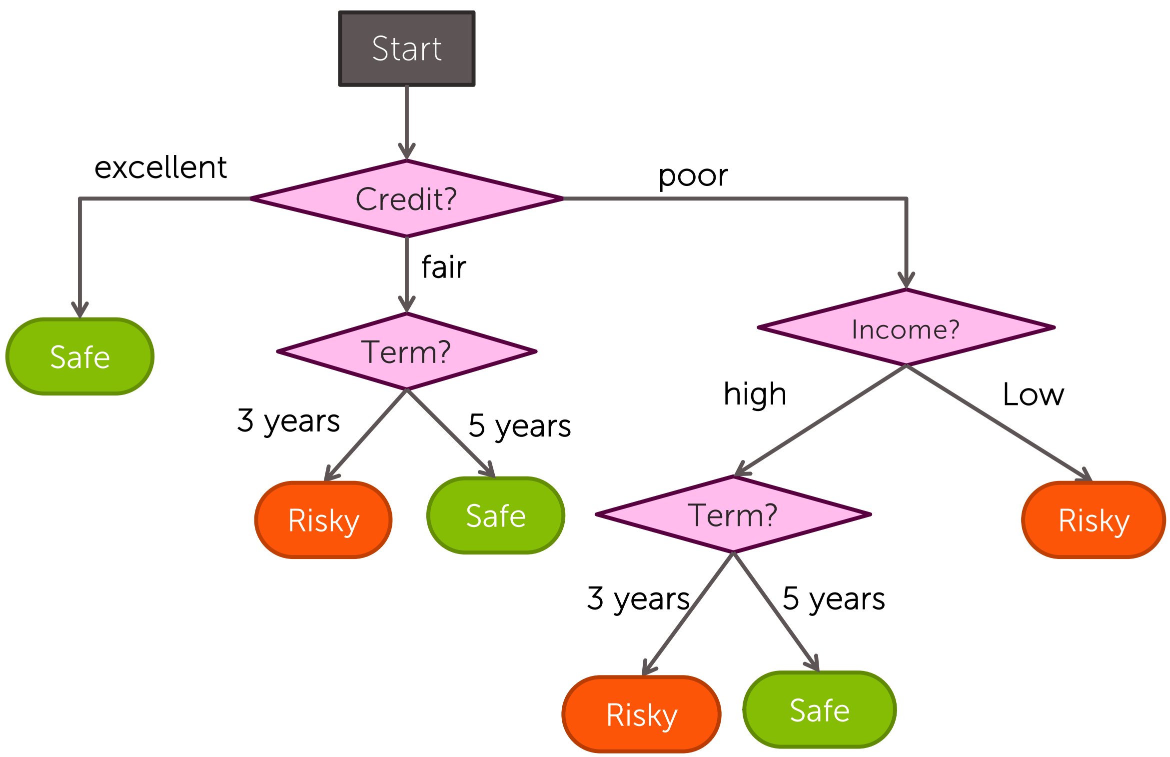
Example: Home loan for a married couple



Classifier review

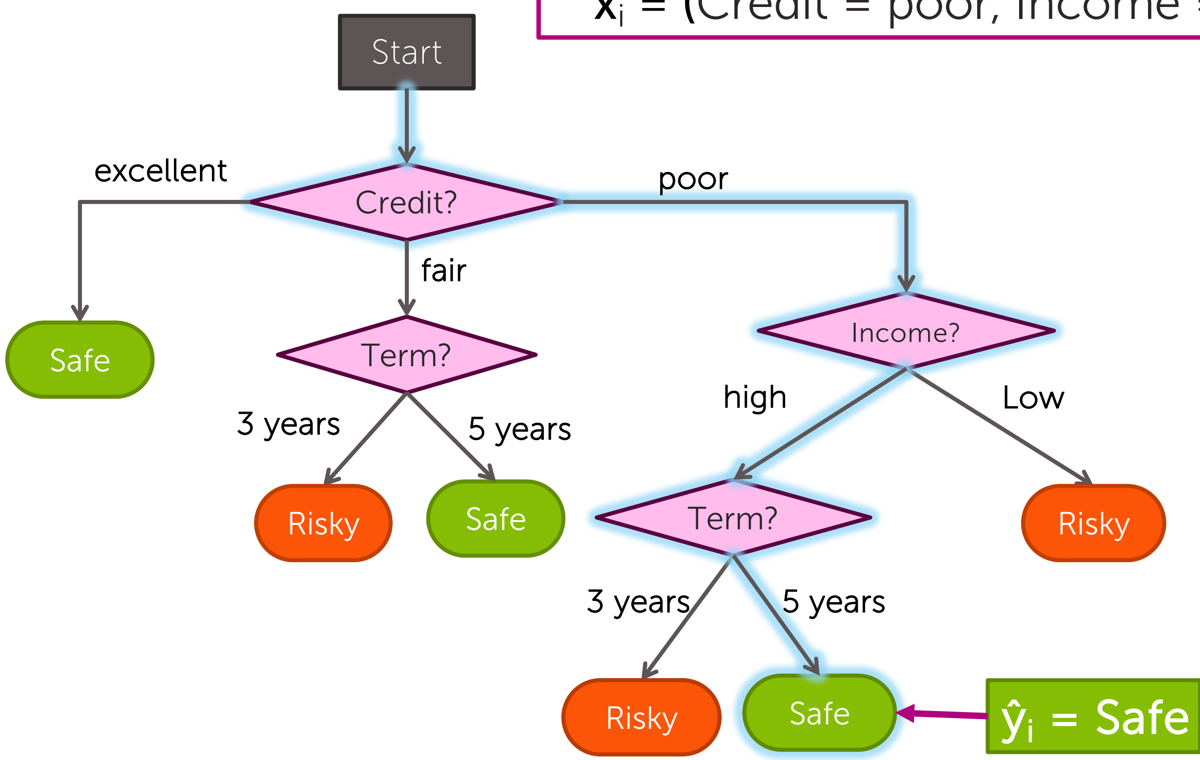


This module … decision trees



Scoring a loan application

$\mathbf{x}_i$ = (Credit = poor, Income = high, Term = 5 years)

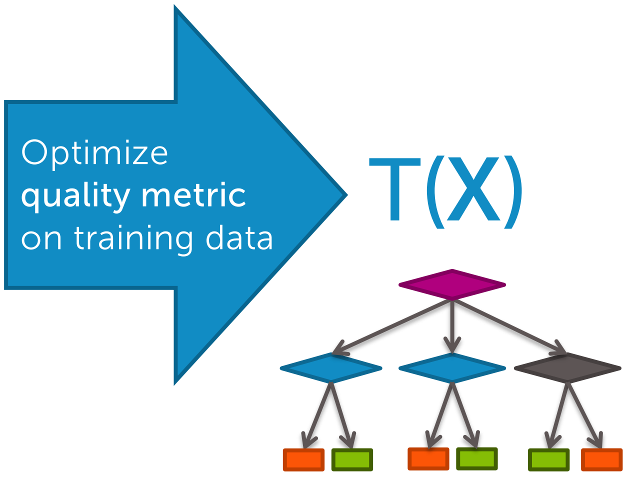


Decision tree learning task

Decision tree learning problem

Training data: N observations ($\mathbf{x}_i$,y_i)

Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe



Quality metric: Classification error

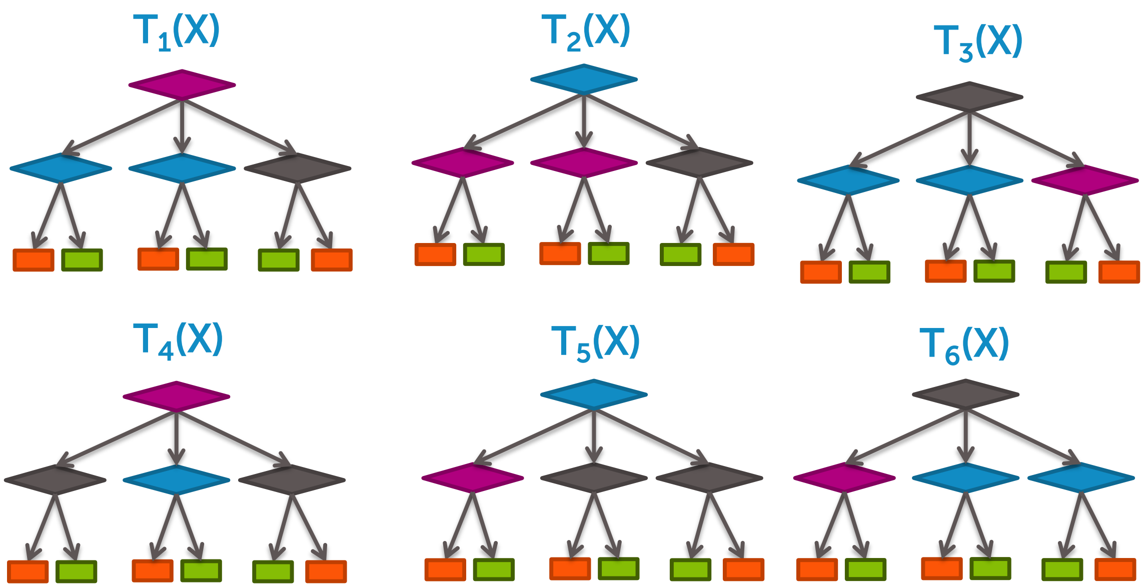
- Error measures fraction of mistakes

$$\text{Error} = \frac{\#\text{ incorrect predictions}}{\#\text{ examples}}$$

  - Best possible value : 0.0
  - Worst possible value: 1.0

How do we find the best tree?

Exponentially large number of possible trees makes decision tree learning hard!



Learning the smallest decision tree is an *NP-hard problem* [Hyafil & Rivest '76]

Greedy decision tree learning

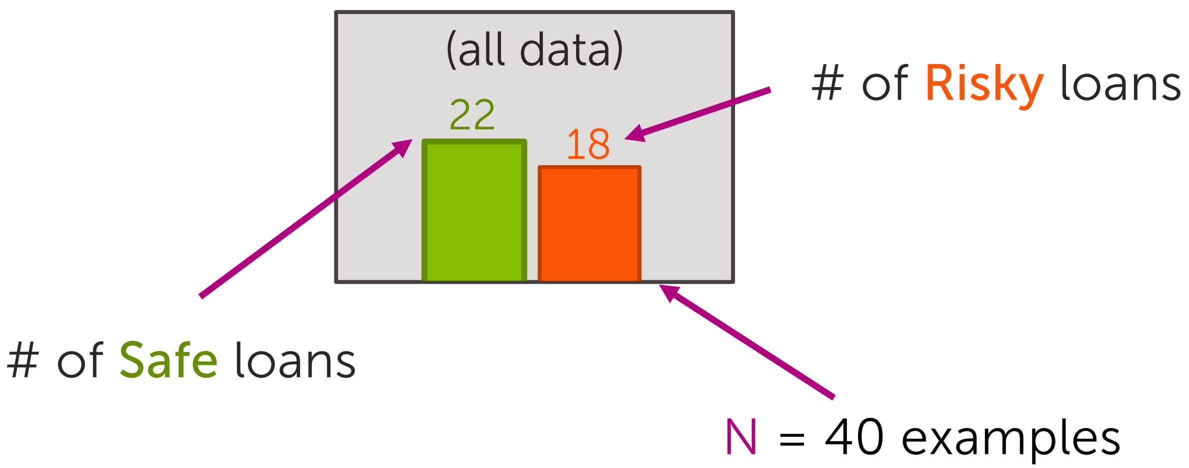
Our training data table

Assume N = 40, 3 features

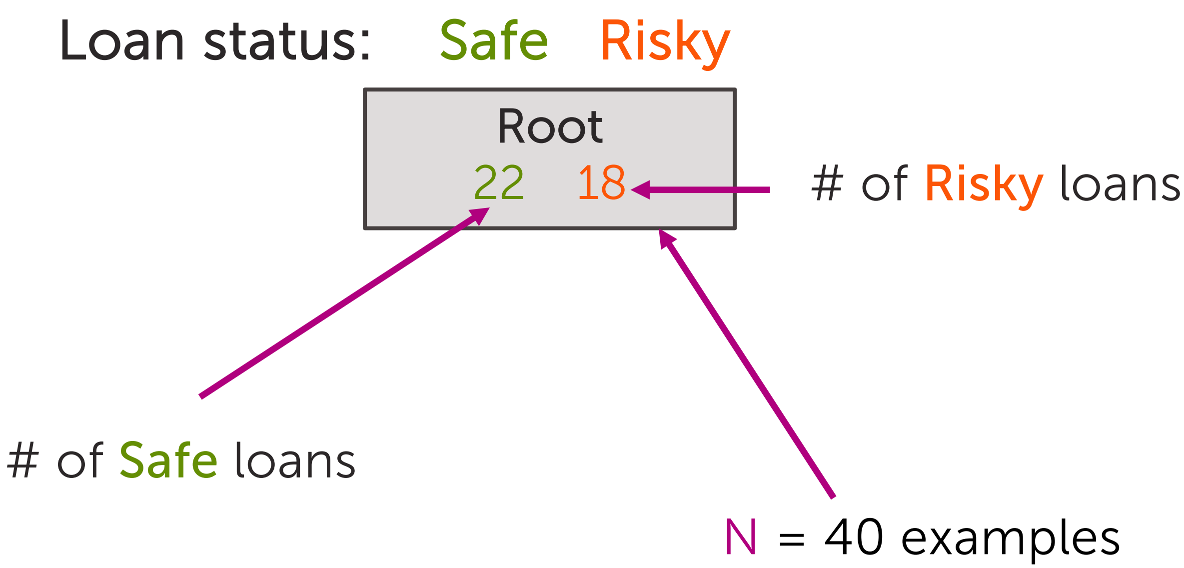
Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe

Start with all the data

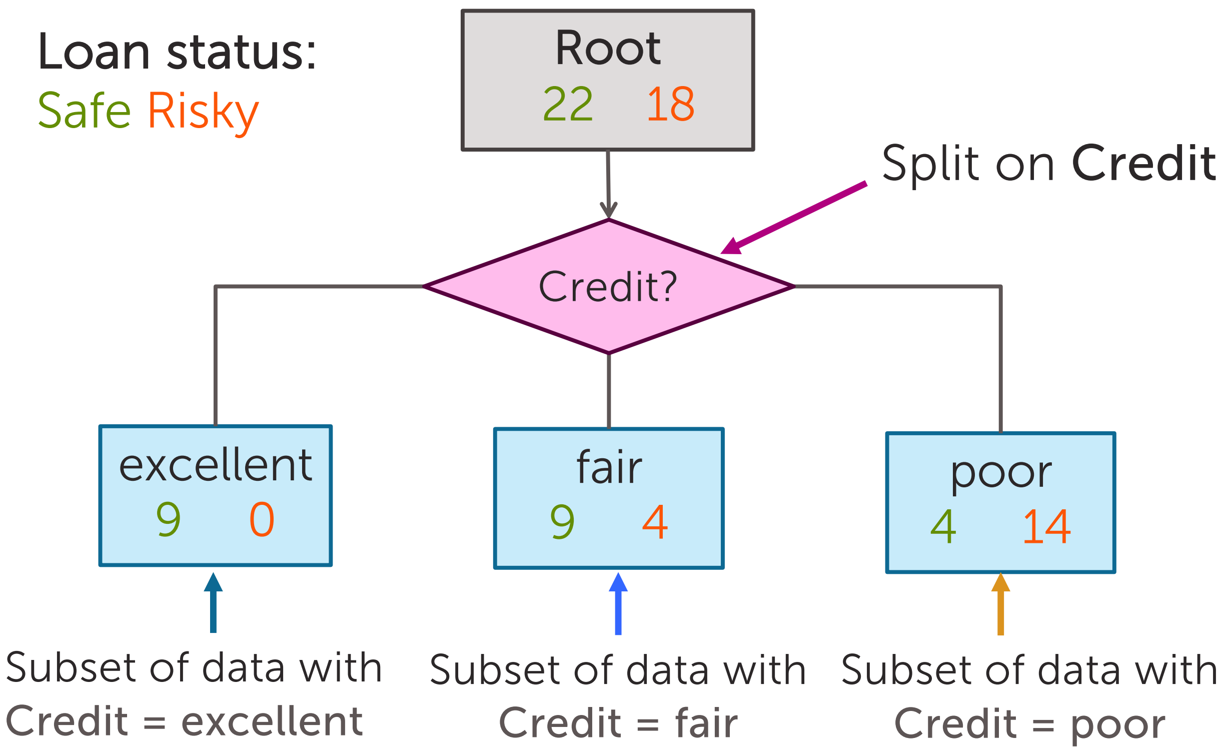
Loan status: Safe Risky



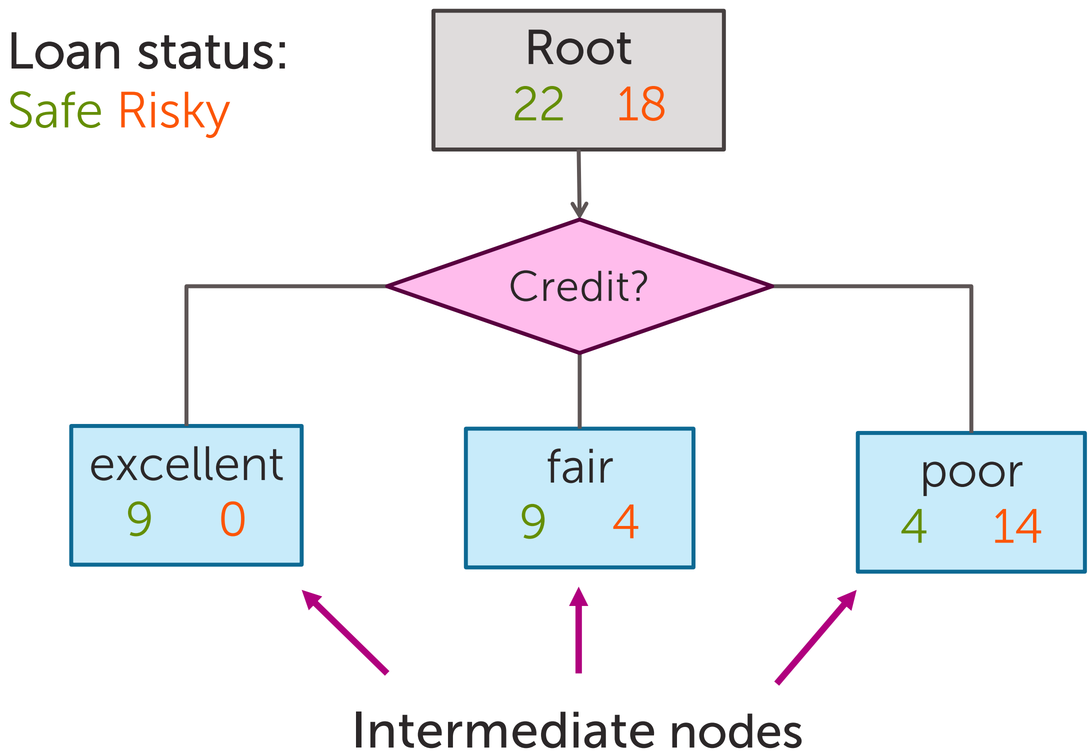
Compact visual notation: Root node



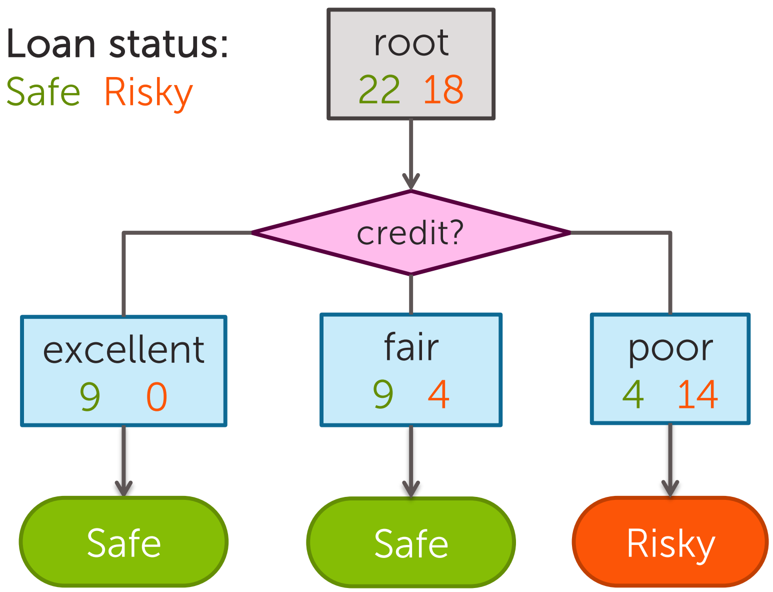
Decision stump: Single level tree



Visual notation: Intermediate nodes



Making predictions with a decision stump

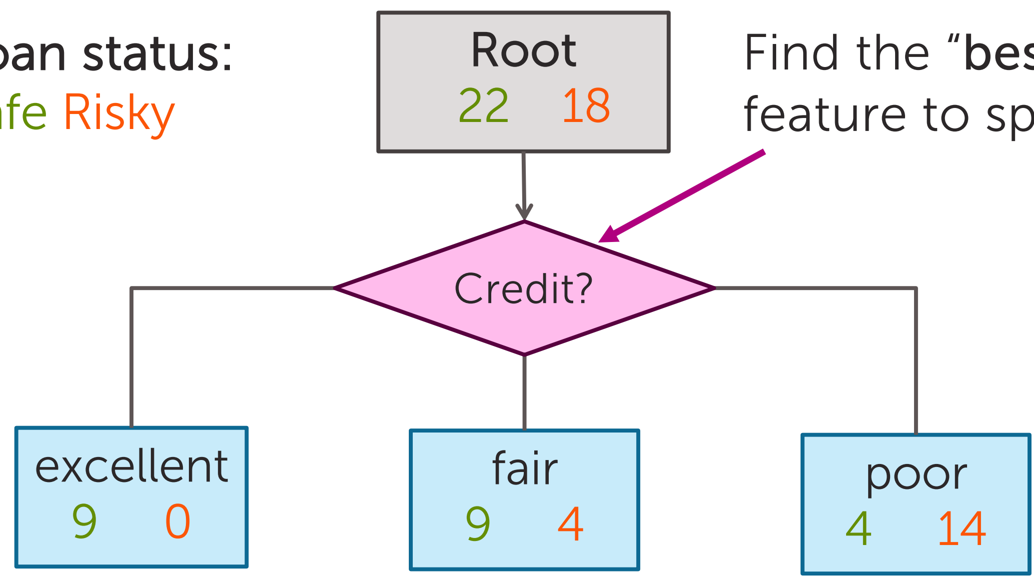


For each intermediate node, set **ŷ = majority value**

Selecting best feature to split on

How do we learn a decision stump?

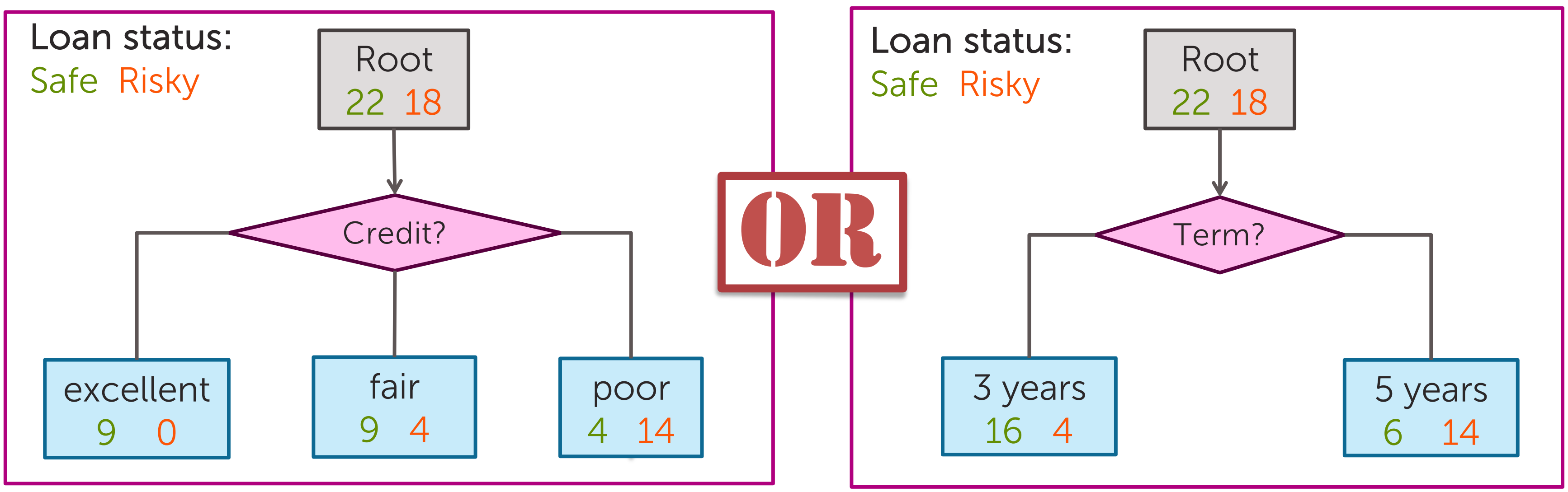
Loan status:
Safe Risky



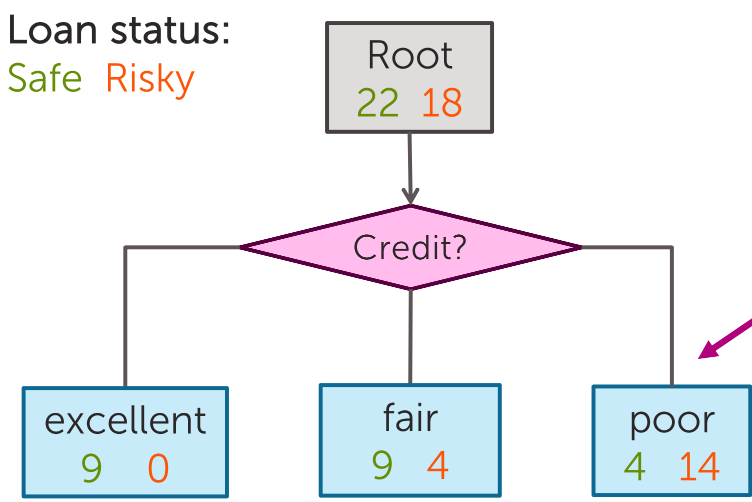
Find the "**best**" feature to split on!

How do we select the best feature?

Choice 1: Split on **Credit** **Choice 2:** Split on **Term**



How do we measure effectiveness of a split?

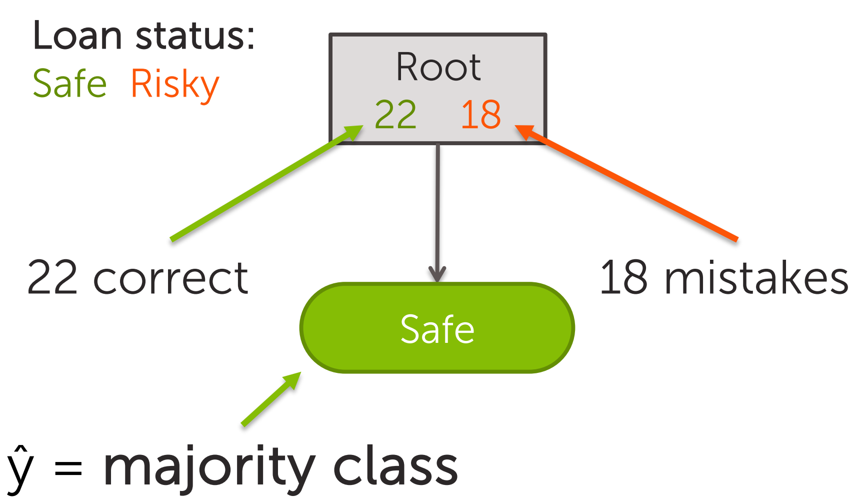


Idea: Calculate classification error of this decision stump

$$\text{Error} = \frac{\text{\# mistakes}}{\text{\# data points}}$$

Calculating classification error

- **Step 1:** ŷ = class of majority of data in node
- **Step 2:** Calculate classification error of predicting ŷ for this data

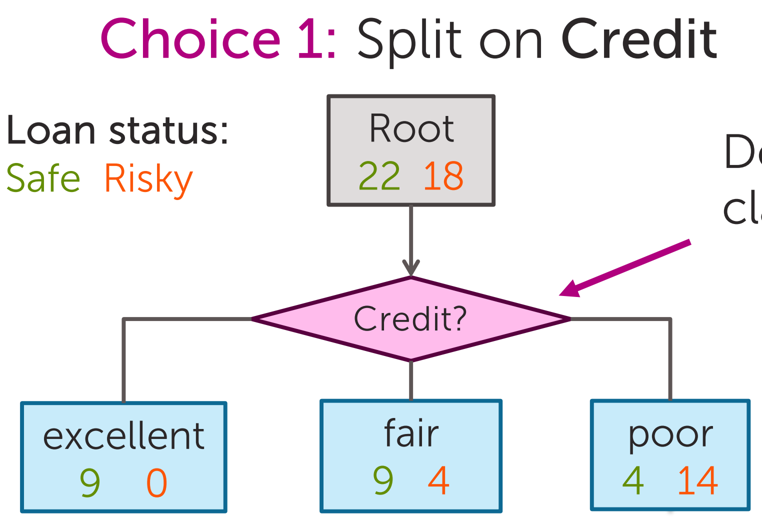


Error = ________

=

Tree	Classification error
(root)	0.45

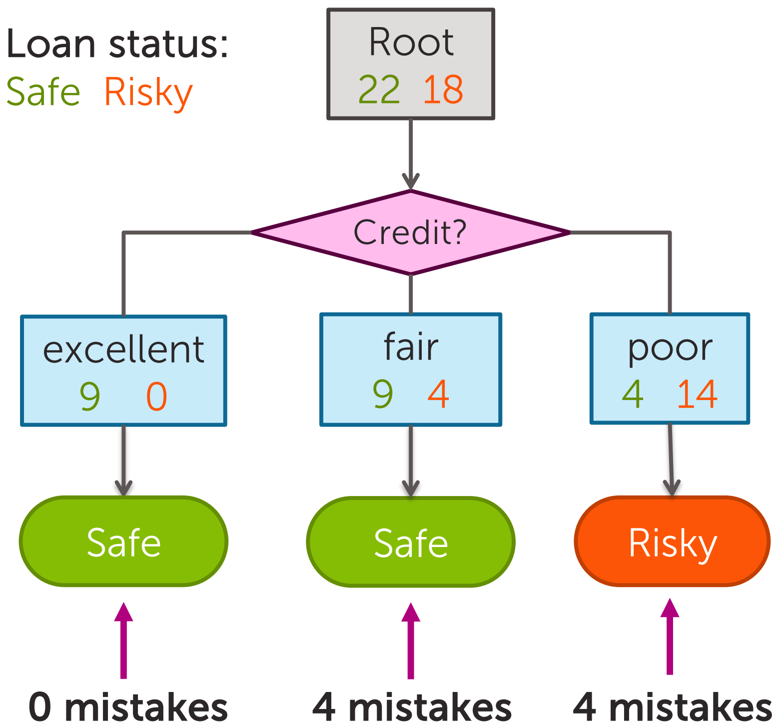
Choice 1: Split on Credit history?



Does a split on Credit reduce classification error below 0.45?

Split on Credit: Classification error

Choice 1: Split on **Credit**

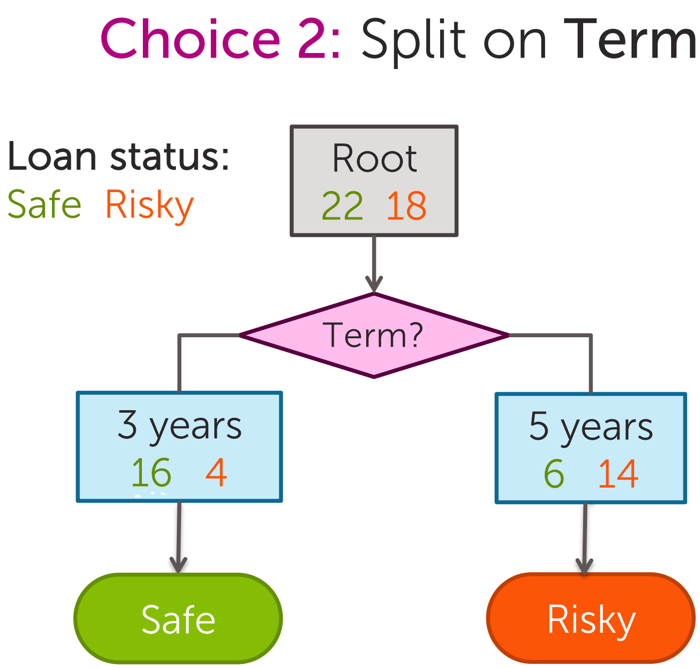


Error = ________

=

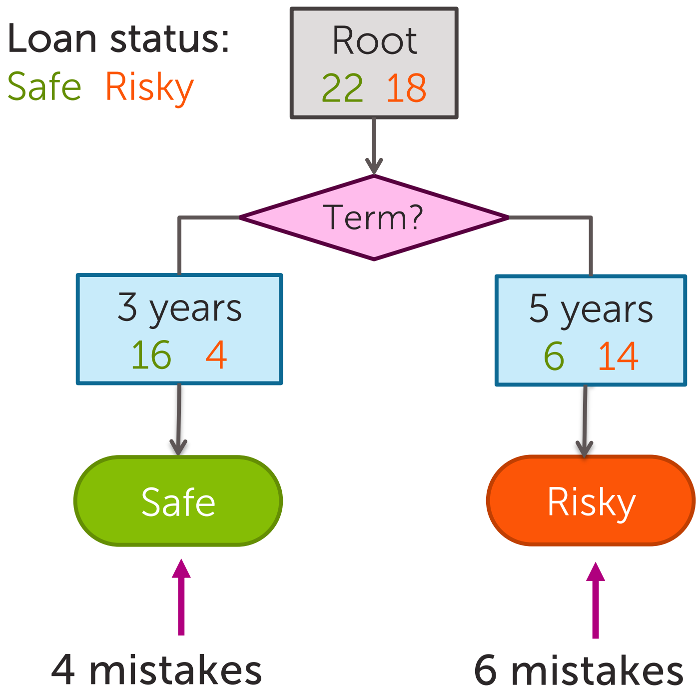
Tree	Classification error
(root)	0.45
Split on **credit**	0.2

Choice 2: Split on Term?



Evaluating the split on Term

Choice 2: Split on **Term**



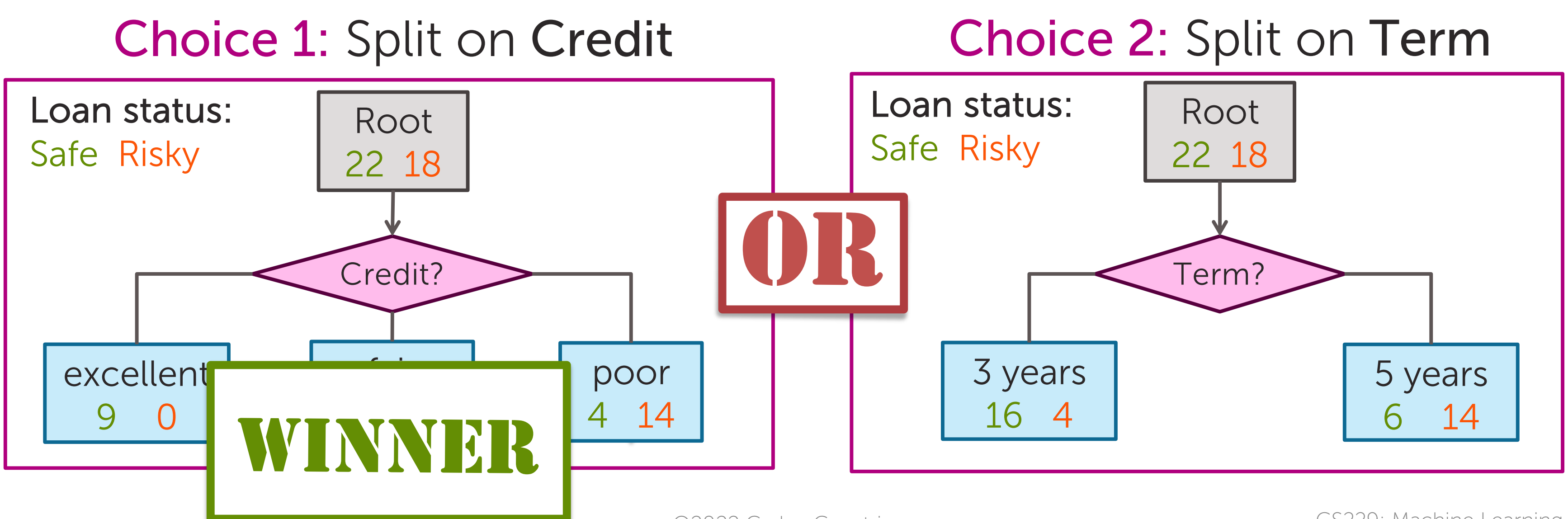
Error = ________

=

Tree	Classification error
(root)	0.45
Split on **credit**	0.2
Split on **term**	0.25

Choice 1 vs Choice 2: Comparing split on Credit vs Term

Tree	Classification error
(root)	0.45
split on **credit**	0.2
split on **loan term**	0.25

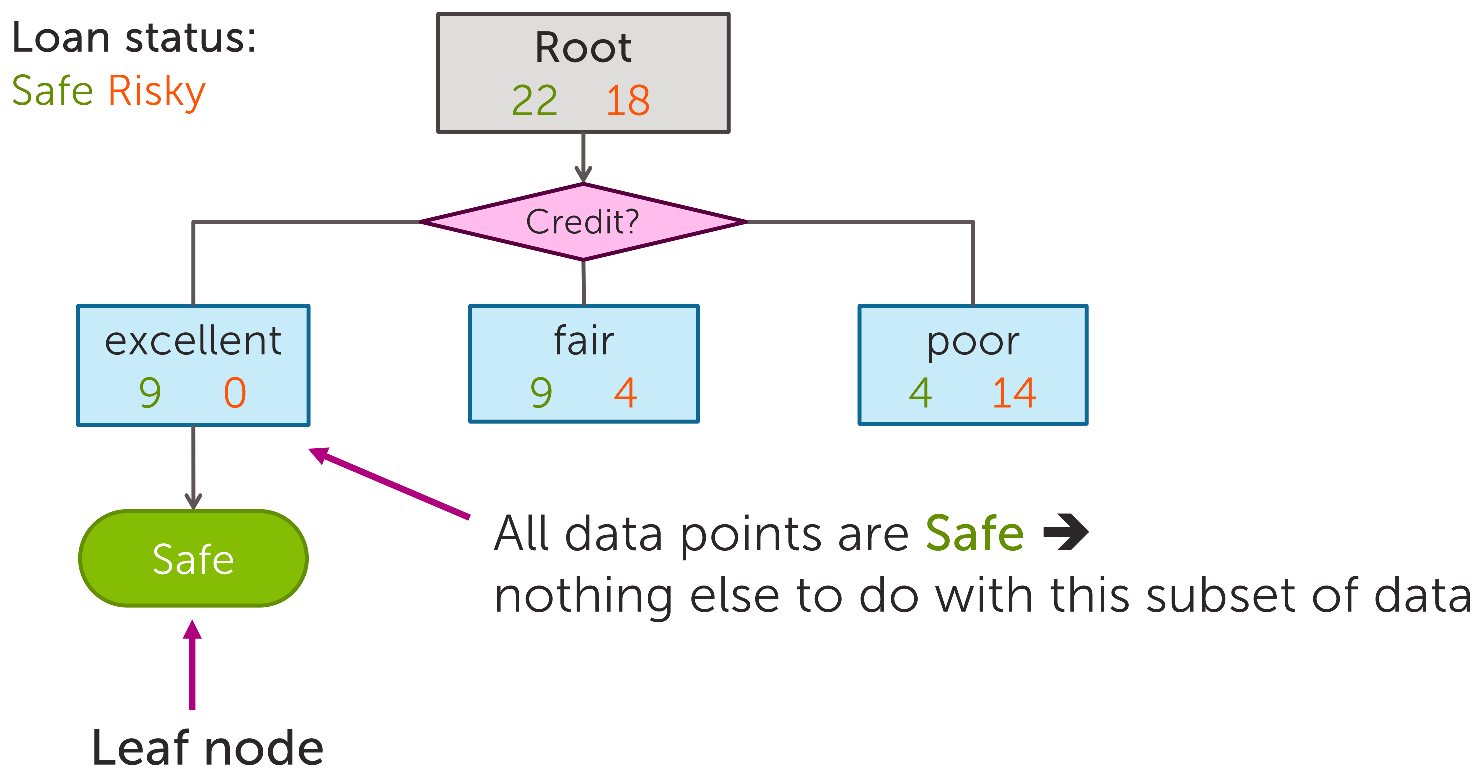


Feature split selection algorithm

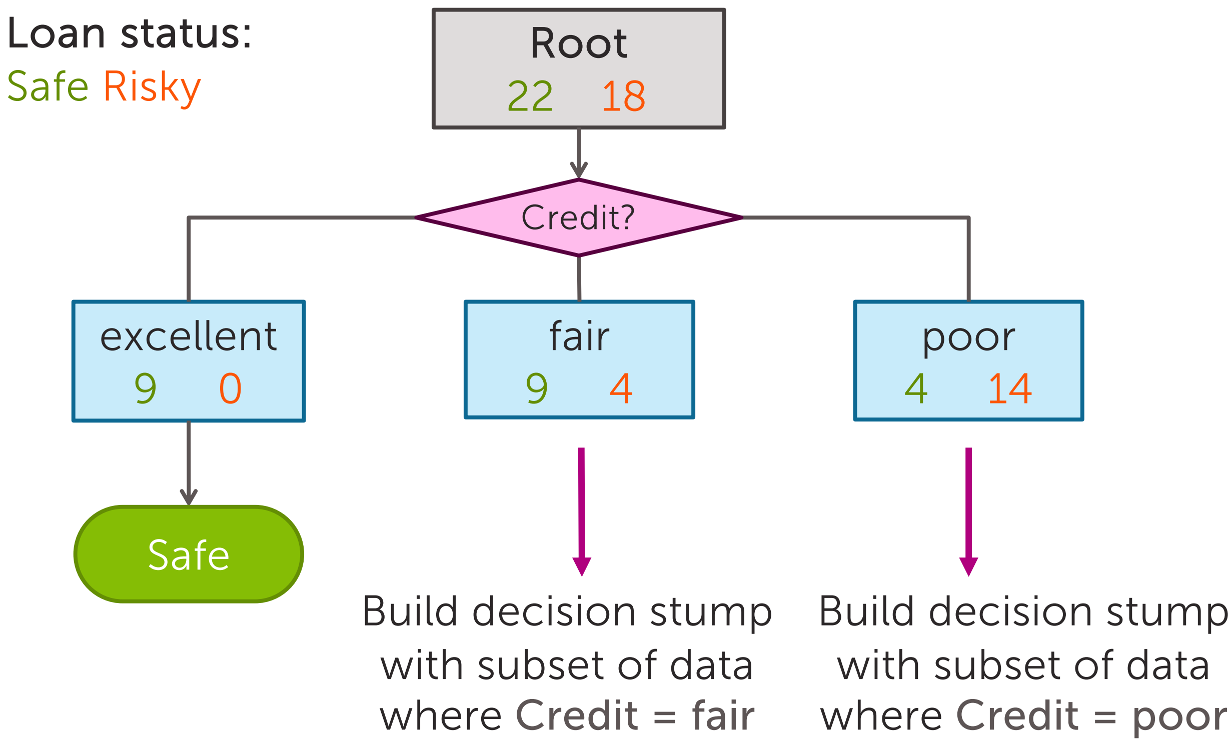
- Given a subset of data M (a node in a tree)
- For each feature $h_i(x)$:
  1. Split data of M according to feature $h_i(x)$
  2. Compute classification error of split
- Chose feature $h^*(x)$ with lowest classification error

Recursion & Stopping conditions

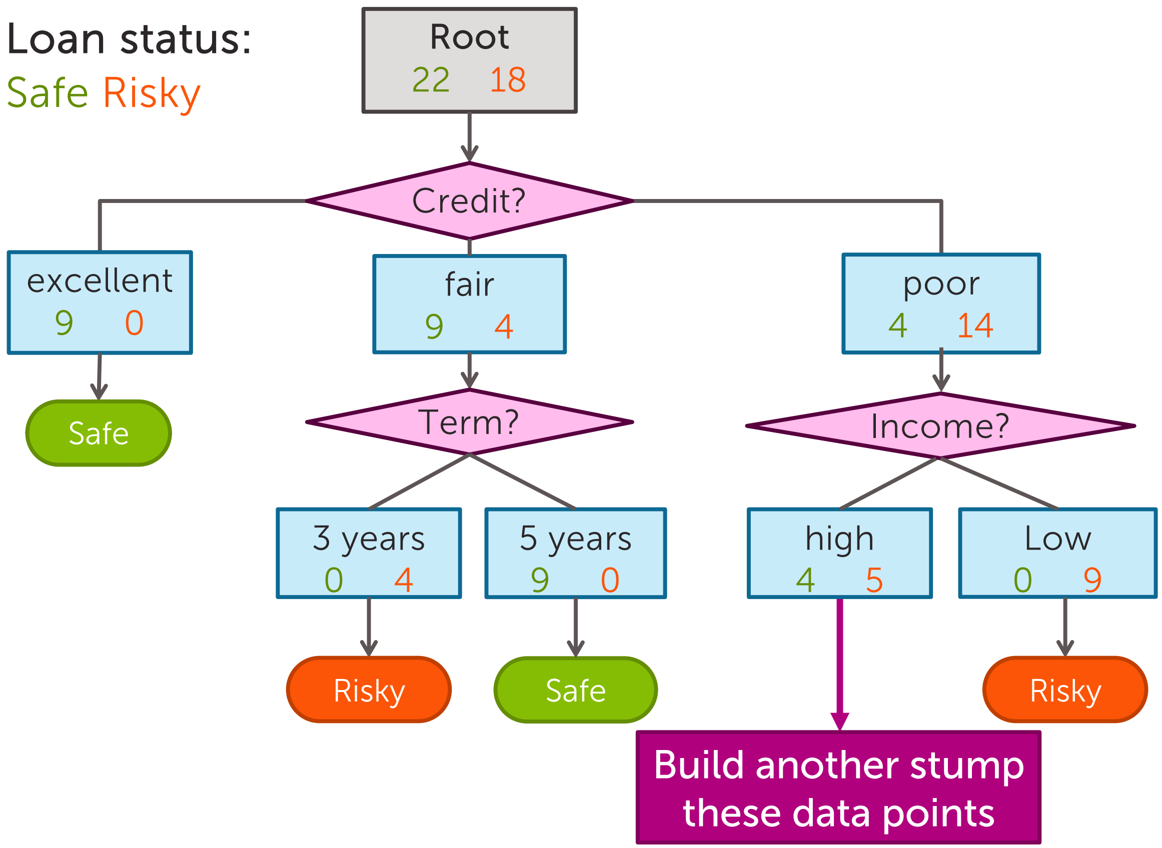
We’ve learned a decision stump, what next?



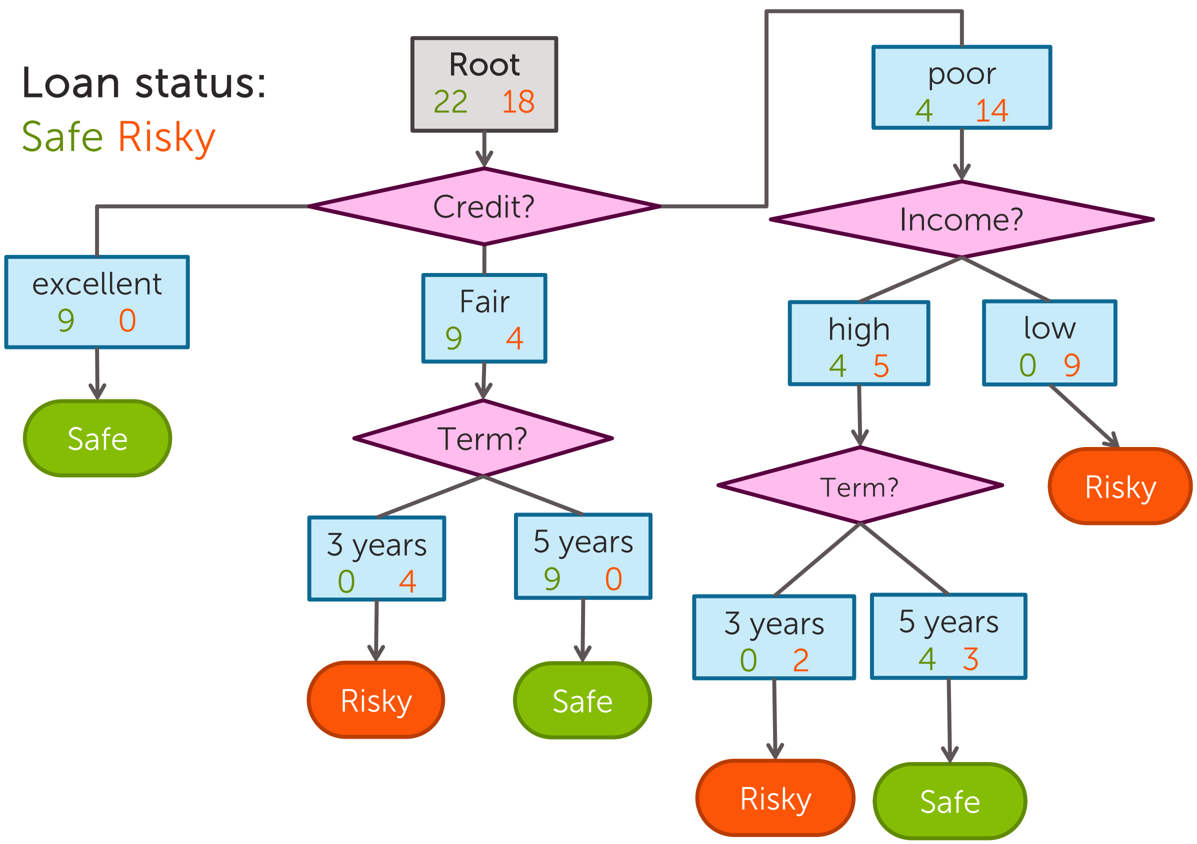
Tree learning = Recursive stump learning



Second level



Final decision tree



Simple greedy decision tree learning

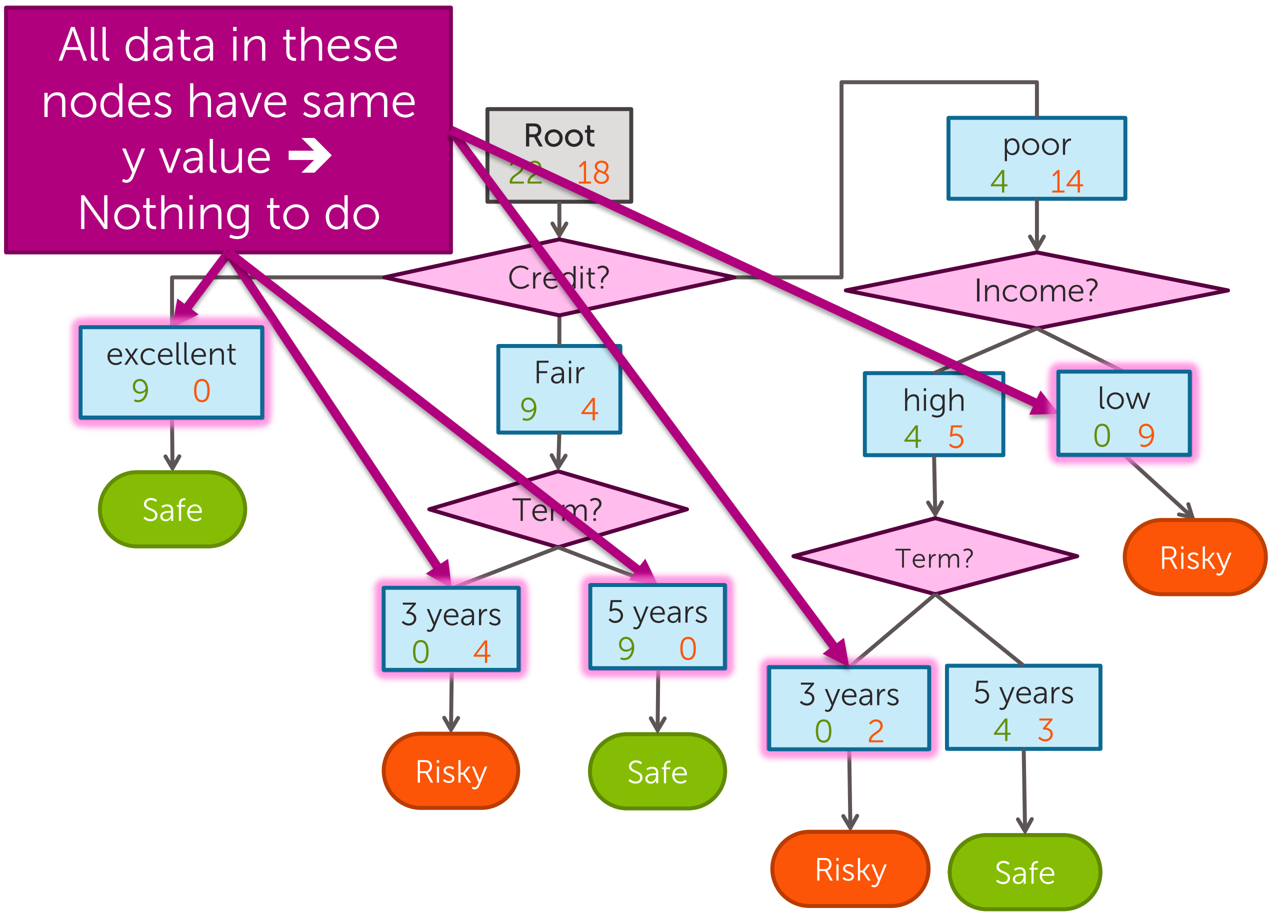
Pick best feature to split on

Learn decision stump with this split

For each leaf of decision stump, recurse

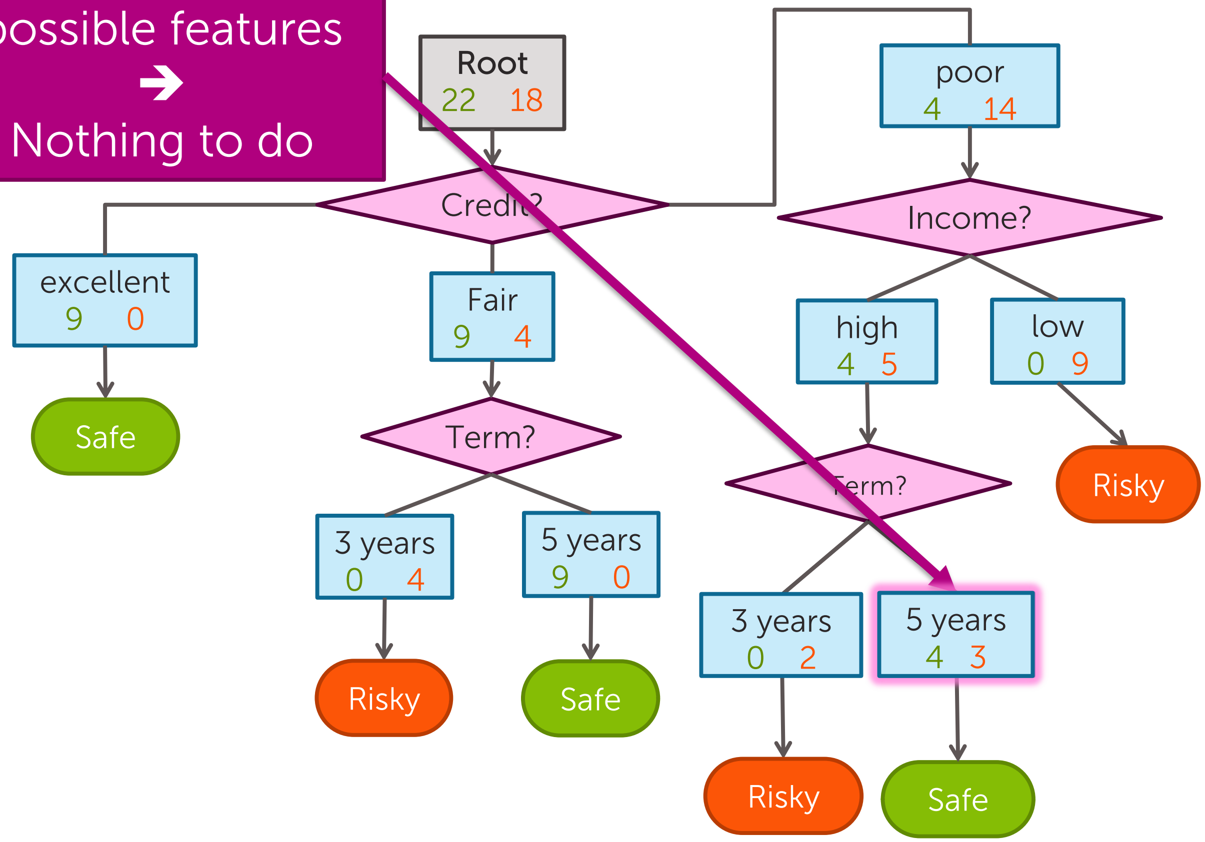
When do we stop???

Stopping condition 1: All data agrees on y



Stopping condition 2: Already split on all features

Already split on all possible features
→
Nothing to do



Greedy decision tree learning

- Step 1: Start with an empty tree
- Step 2: Select a feature to split data
- For each split of the tree:
  - Step 3: If nothing more to do, make predictions
  - Step 4: Otherwise, go to Step 2 & continue (recurse) on this split

Pick feature split leading to lowest classification error

Stopping conditions

Recursion

Is this a good idea?

Proposed stopping condition 3:
Stop if no split reduces the classification error

Stopping condition 3:
Don't stop if error doesn't decrease???

y = **x**[1] xor **x**[2]

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False

y values
True False

Root
2 2

Error = ________

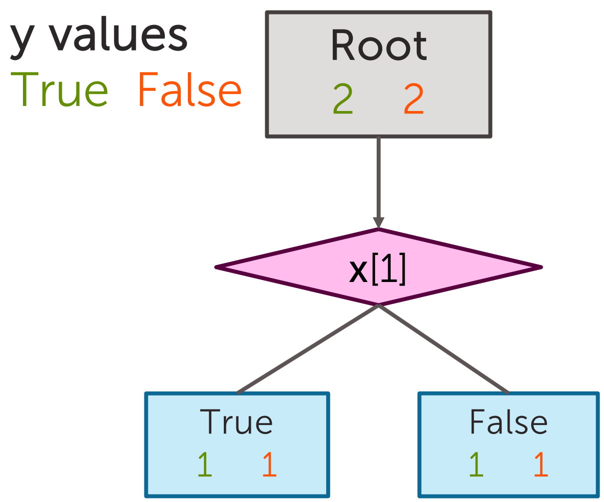
=

Tree	Classification error
(root)	0.5

Consider split on x[1]

y = **x**[1] xor **x**[2]

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False



Error = ________

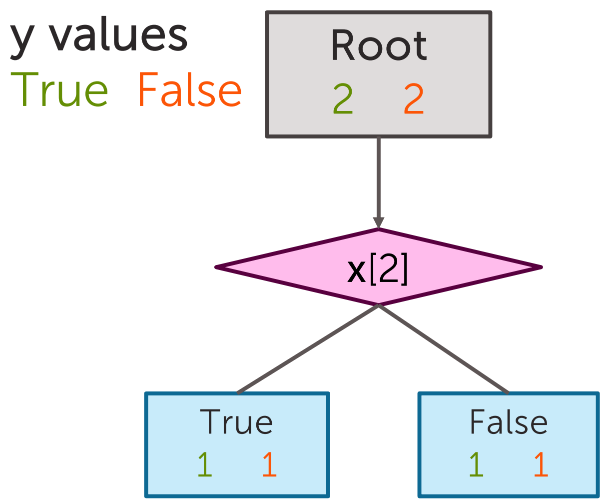
=

Tree	Classification error
(root)	0.5
Split on **x**[1]	0.5

Consider split on x[2]

$y = \mathbf{x}[1]$ xor $\mathbf{x}[2]$

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False



$$\text{Error} = \frac{1+1}{2+2} = 0.5$$

Neither features improve training error… Stop now???

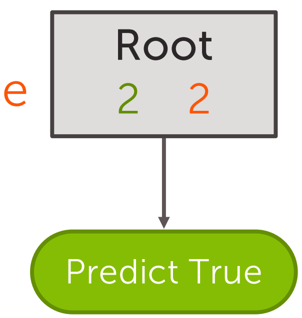
Tree	Classification error
(root)	0.5
Split on **x**[1]	0.5
Split on **x**[2]	0.5

Final tree with stopping condition 3

$y = \mathbf{x}[1]$ xor $\mathbf{x}[2]$

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False

y values
True False



Tree	Classification error
with stopping condition 3	0.5

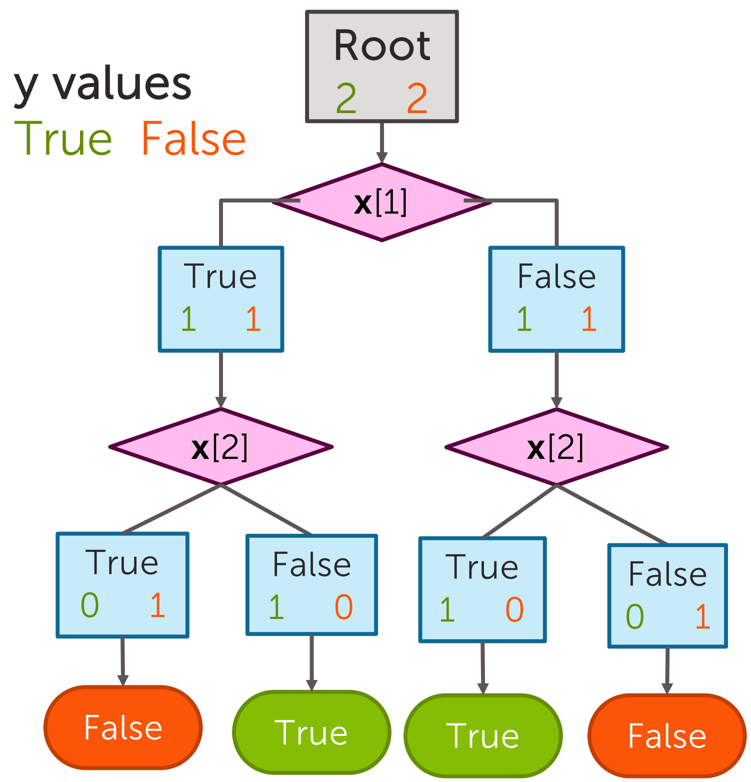
Without stopping condition 3

Condition 3 (stopping when training error doesn't' improve) is **not** recommended!

y = **x**[1] xor **x**[2]

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False

Tree	Classification error
with stopping condition 3	0.5
without stopping condition 3	

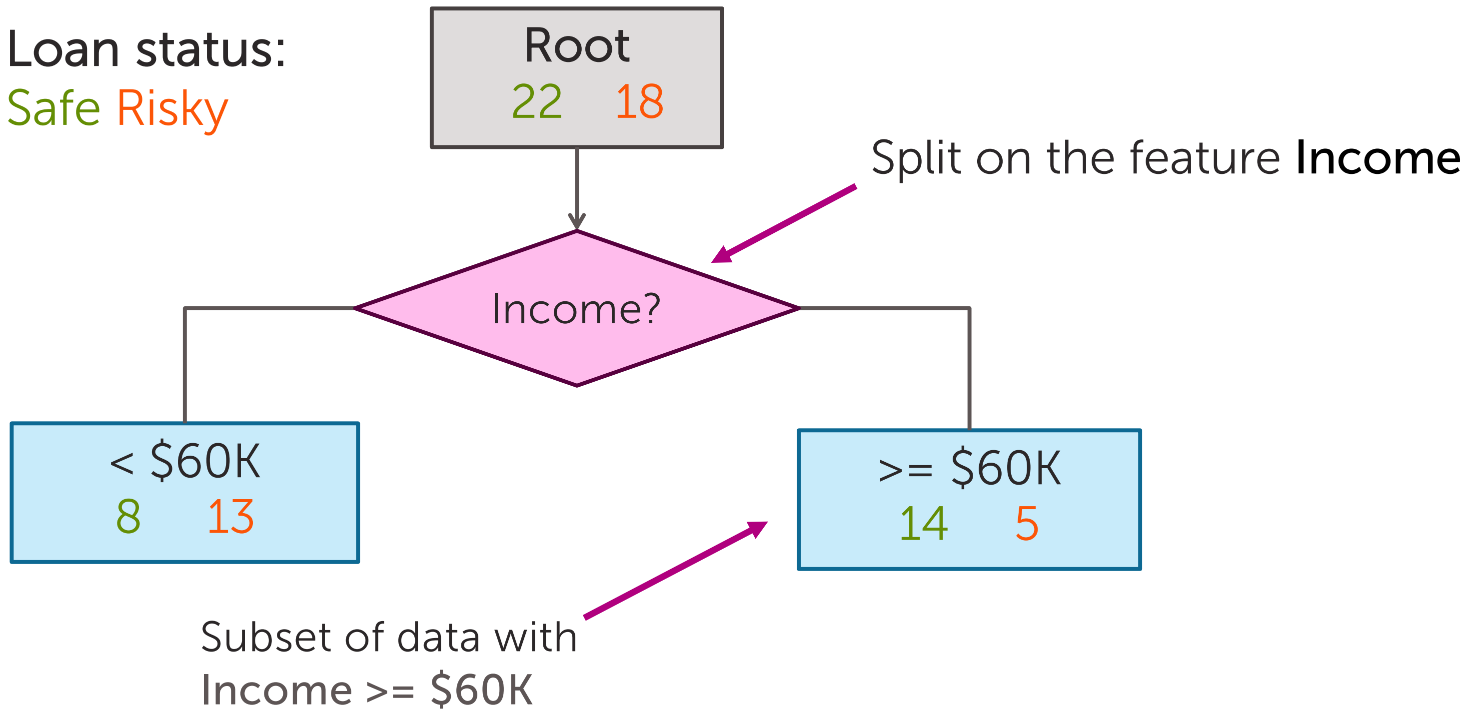


Decision tree learning:
Real valued features

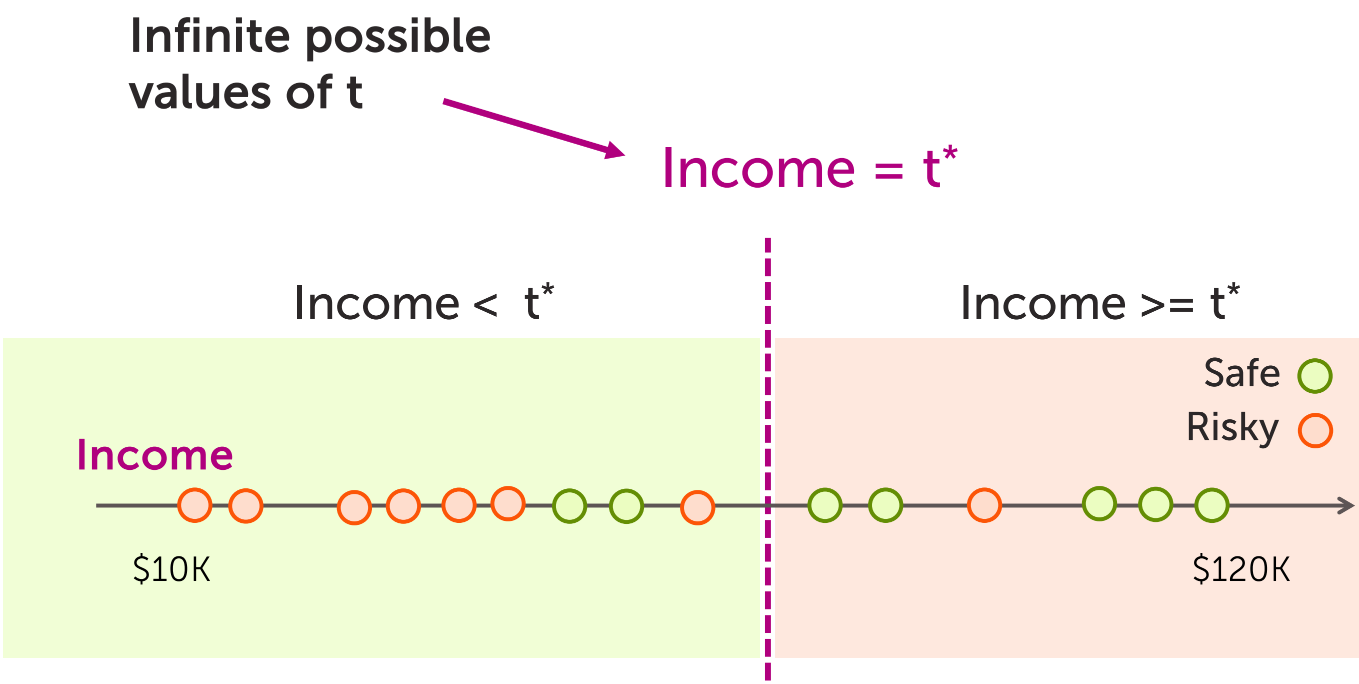
How do we use real values inputs?

Income	Credit	Term	y
$105 K	excellent	3 yrs	Safe
$112 K	good	5 yrs	Risky
$73 K	fair	3 yrs	Safe
$69 K	excellent	5 yrs	Safe
$217 K	excellent	3 yrs	Risky
$120 K	good	5 yrs	Safe
$64 K	fair	3 yrs	Risky
$340 K	excellent	5 yrs	Safe
$60 K	good	3 yrs	Risky

Threshold split

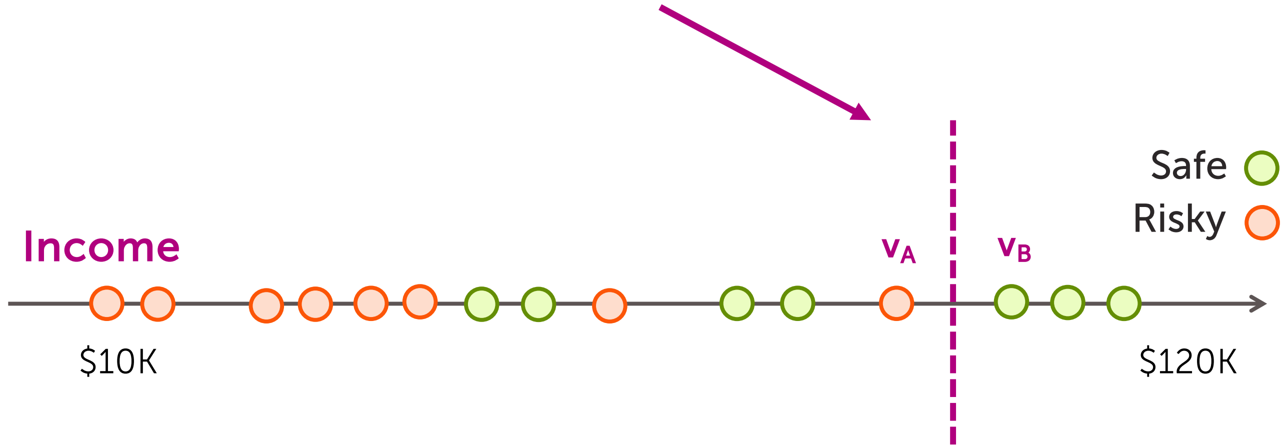


Finding the best threshold split



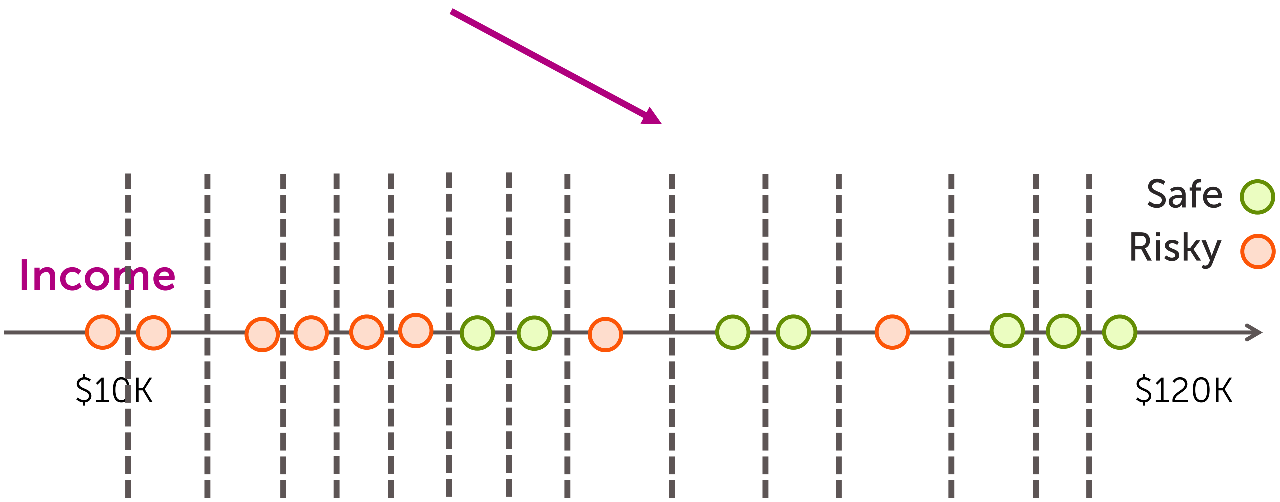
Consider a threshold between points

Same classification error for any threshold split between v_A and v_B



Only need to consider mid-points

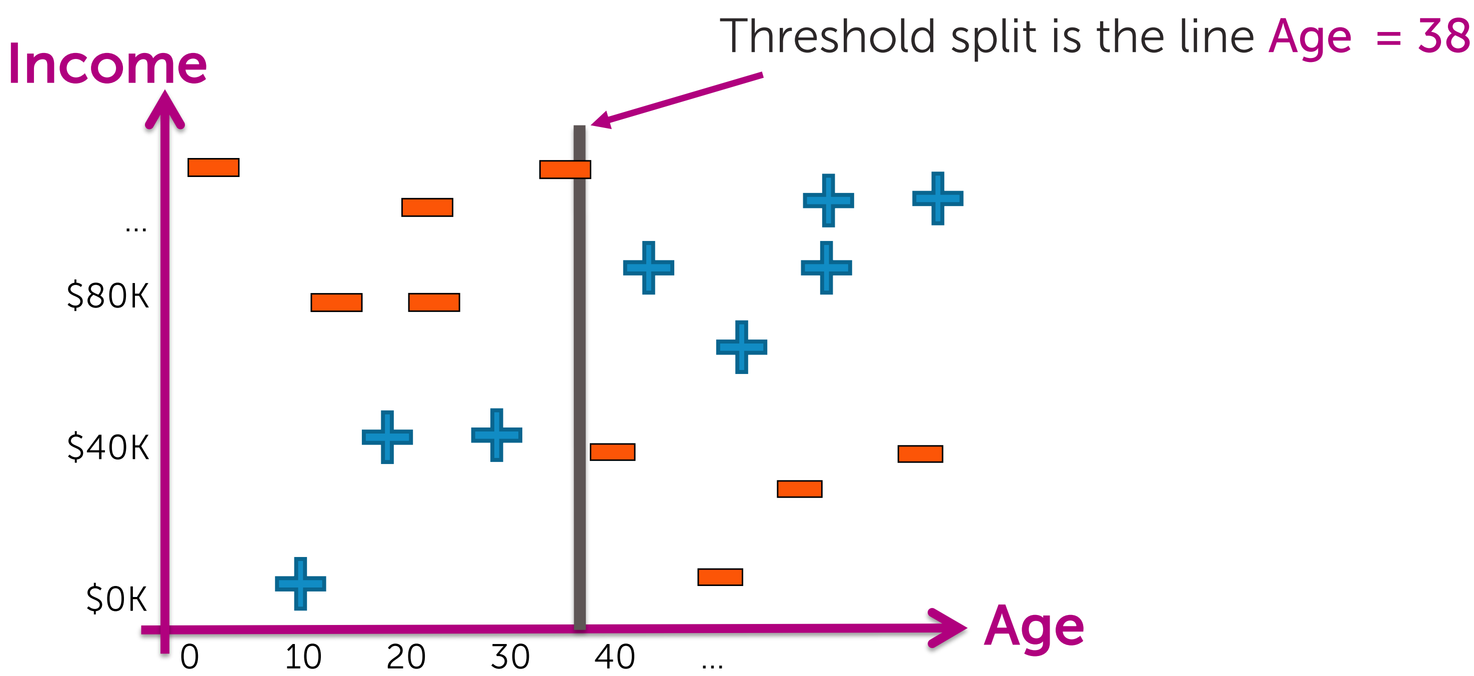
Finite number of
splits to consider



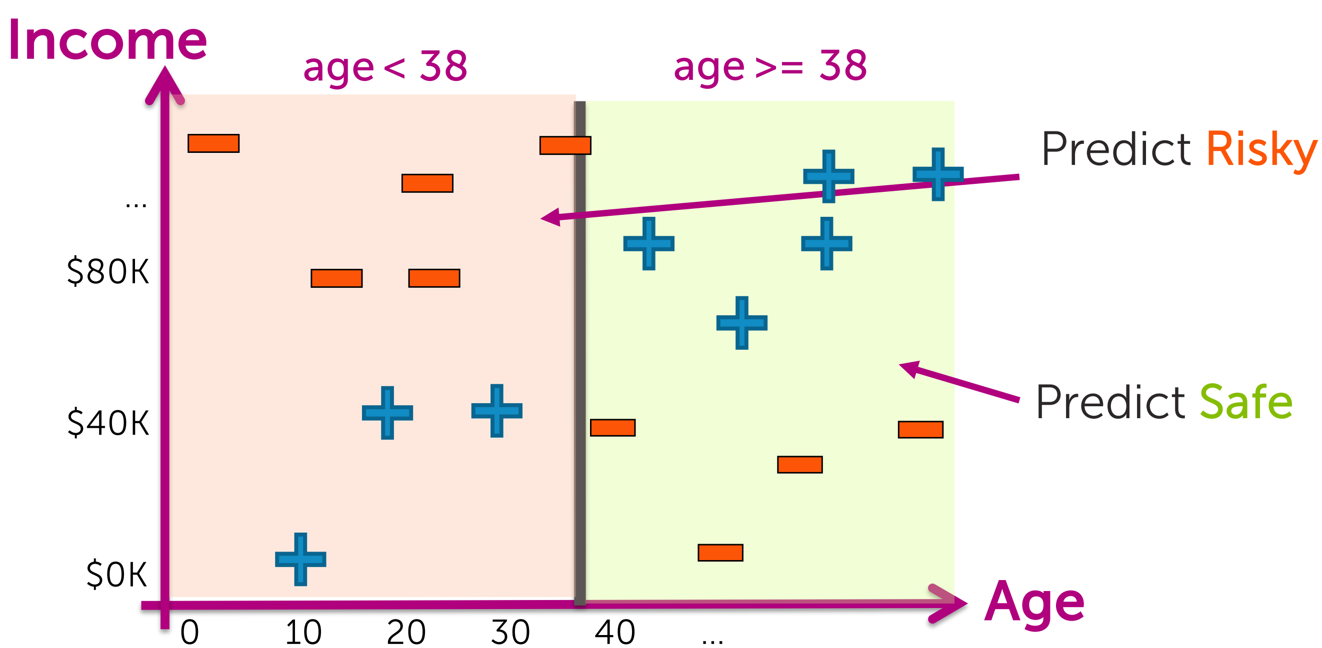
Threshold split selection algorithm

- Step 1: Sort the values of a feature $h_j(\mathbf{x})$:
  Let $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots \mathbf{v}_N\}$ denote sorted values
- Step 2:
  - For i = 1 … N-1
    - Consider split $t_i = (\mathbf{v}_i + \mathbf{v}_{i+1}) / 2$
    - Compute classification error for treshold split $h_j(\mathbf{x}) >= \mathbf{t}_i$
  - Chose the $\mathbf{t}^*$ with the lowest classification error

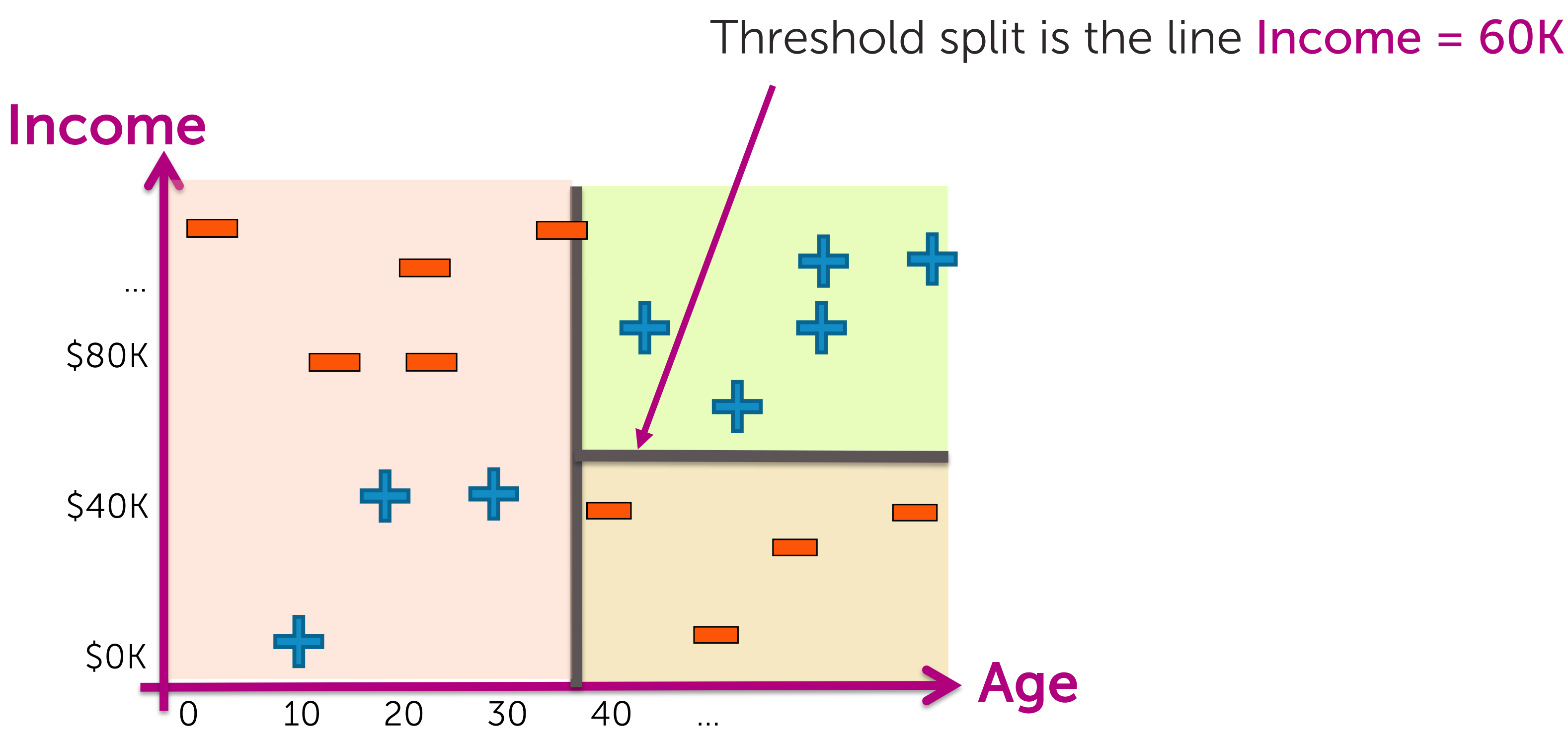
Visualizing the threshold split



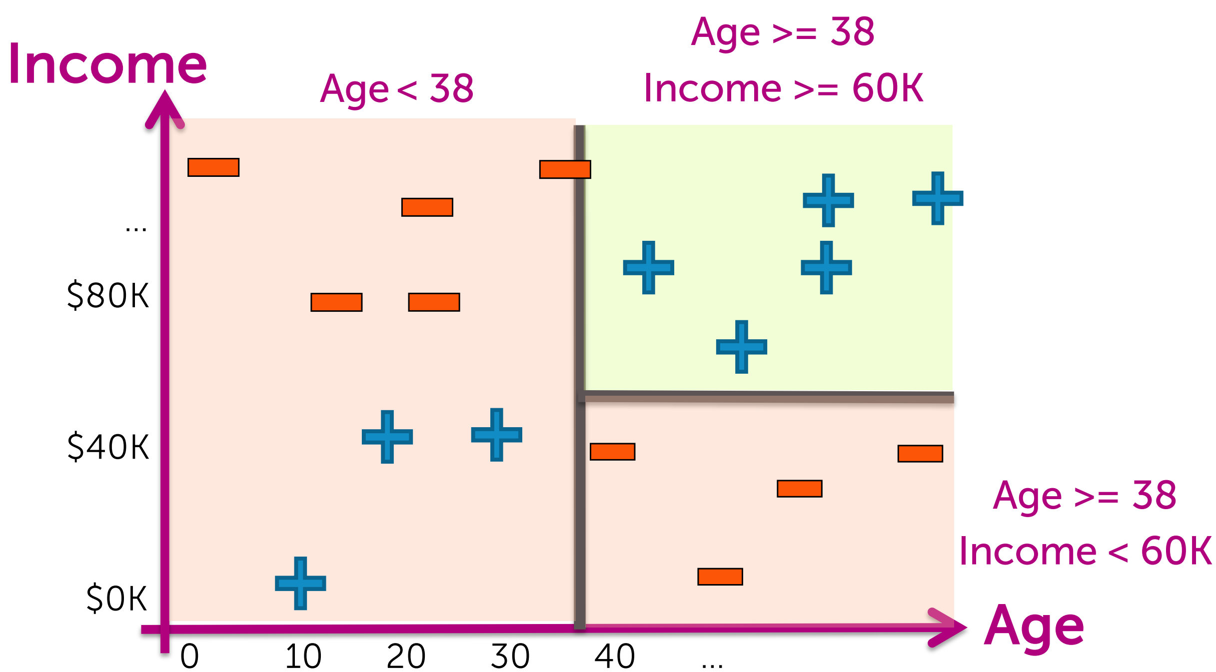
Split on Age >= 38



Depth 2: Split on Income >= $60K



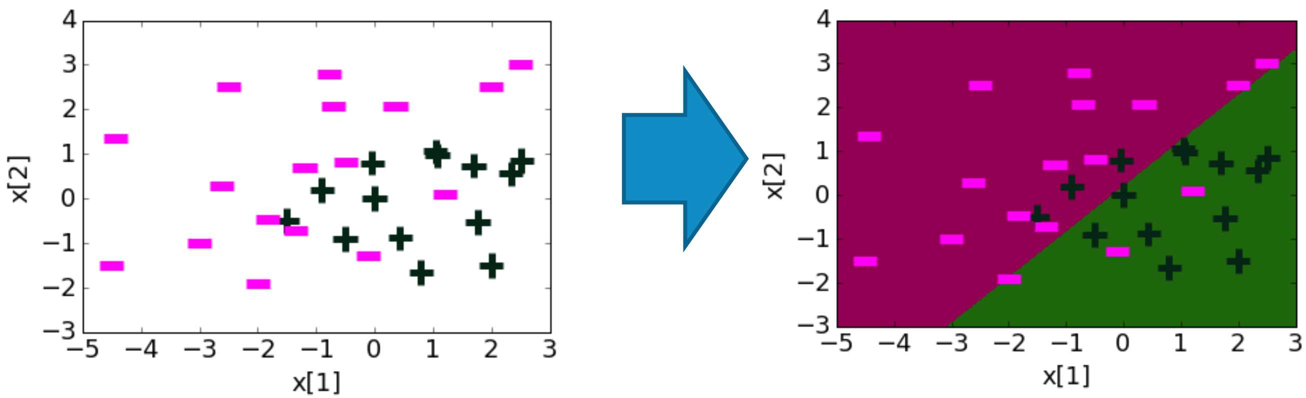
Each split partitions the 2-D space



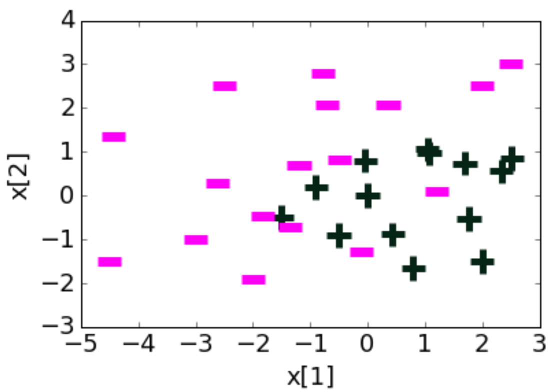
Decision trees vs logistic regression: *Example*

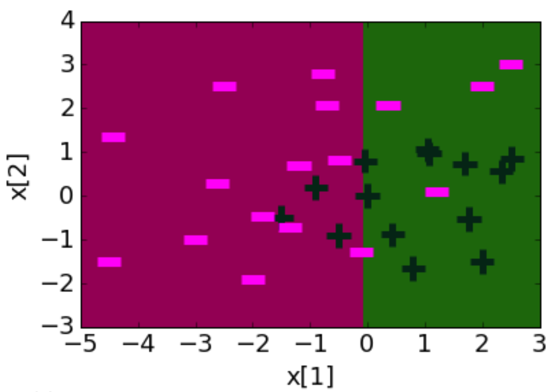
Logistic regression

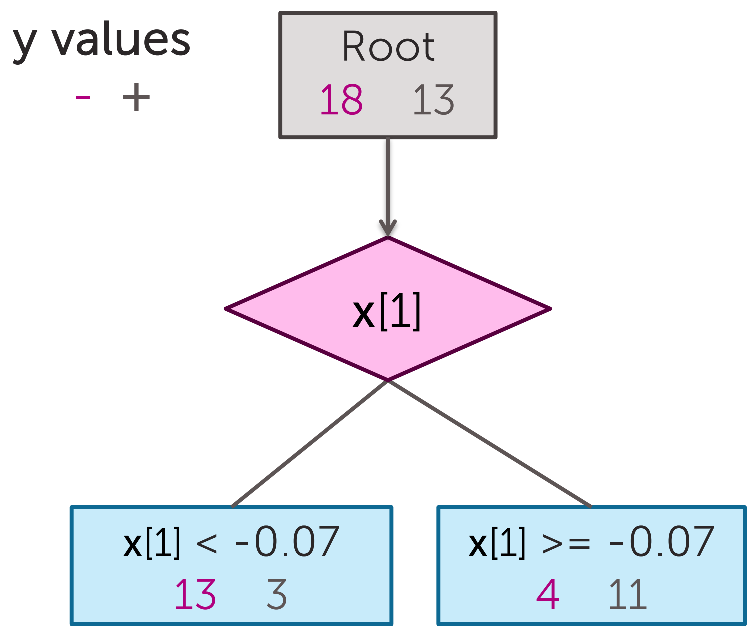
Feature	Value	Weight Learned
$h_0(\mathbf{x})$	1	0.22
$h_1(\mathbf{x})$	x[1]	1.12
$h_2(\mathbf{x})$	x[2]	-1.07



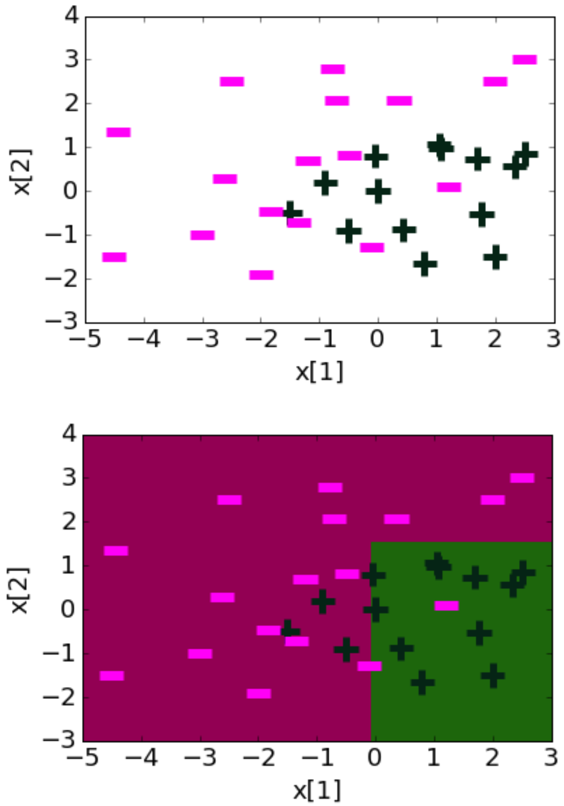
Depth 1: Split on x[1]

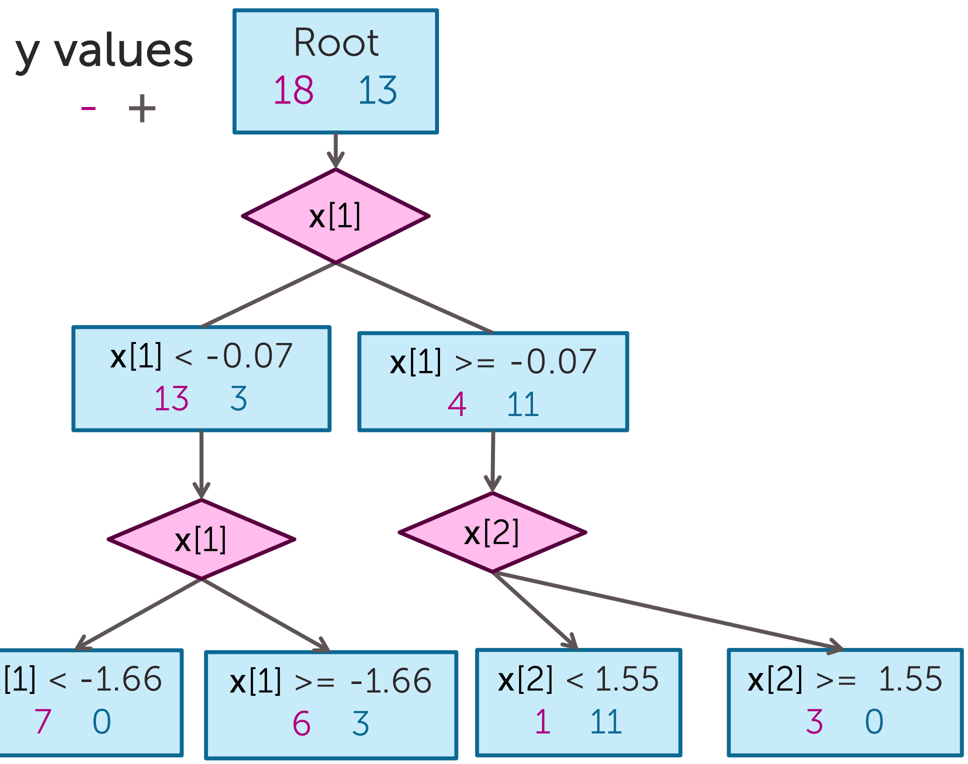






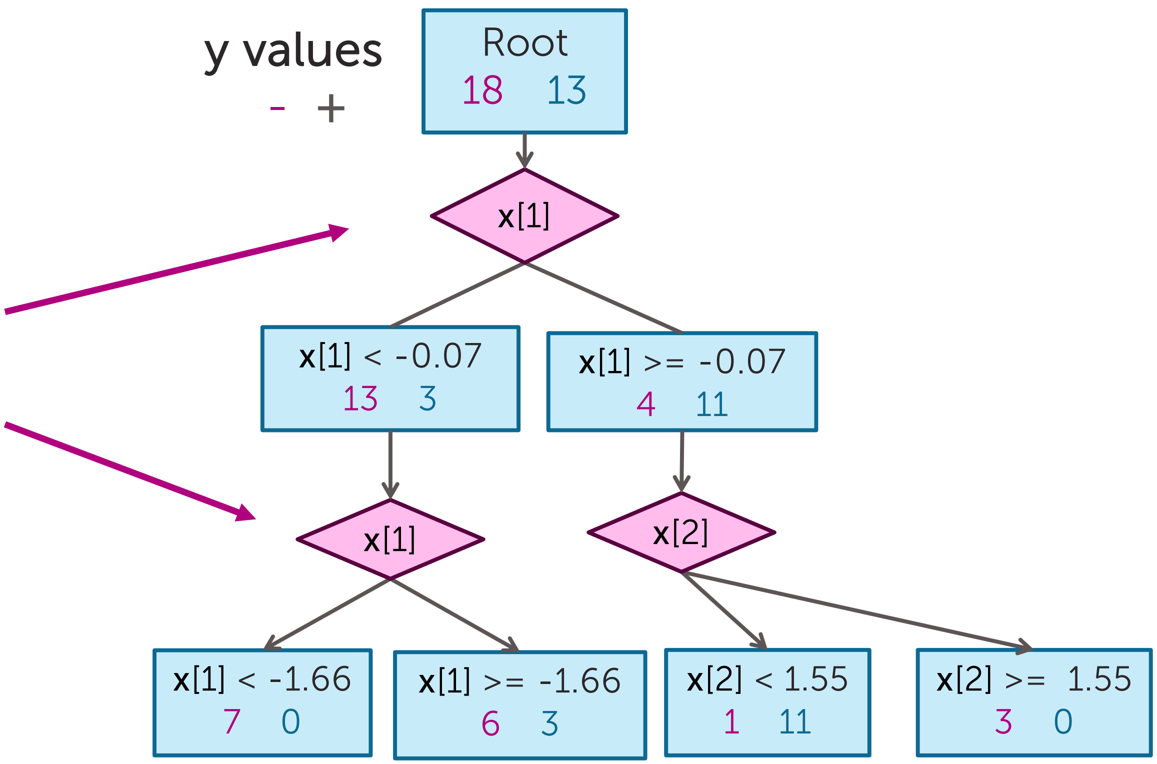
Depth 2


x[2]
x[1]
4
3
2
1
0
−1
−2
−3
−5 −4 −3 −2 −1 0 1 2 3

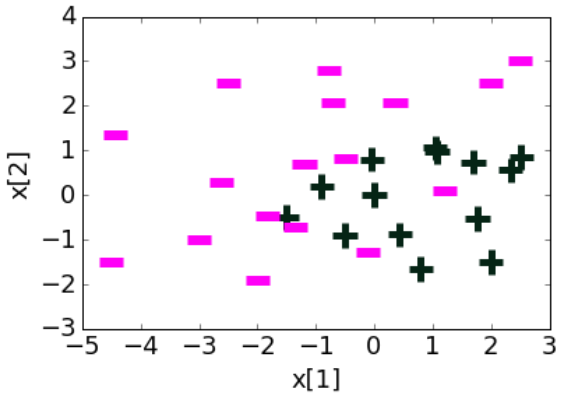

y values
- +
Root
18 13
x[1]
x[1] < -0.07
13 3
x[1] >= -0.07
4 11
x[1]
x[2]
x[1] < -1.66
7 0
x[1] >= -1.66
6 3
x[2] < 1.55
1 11
x[2] >= 1.55
3 0

Threshold split caveat

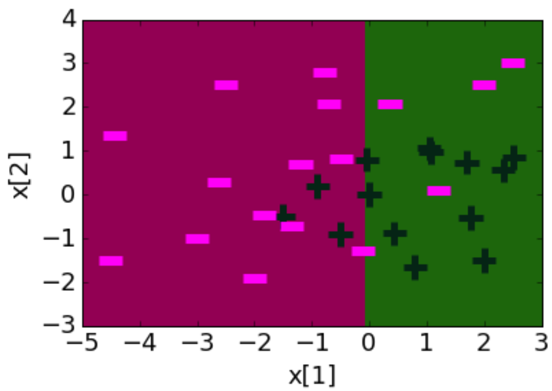
For threshold splits, same feature can be used multiple times



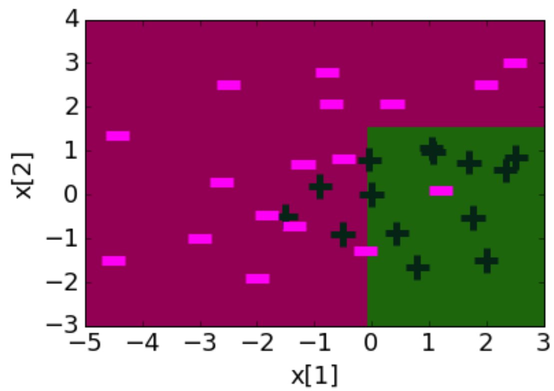
Decision boundaries



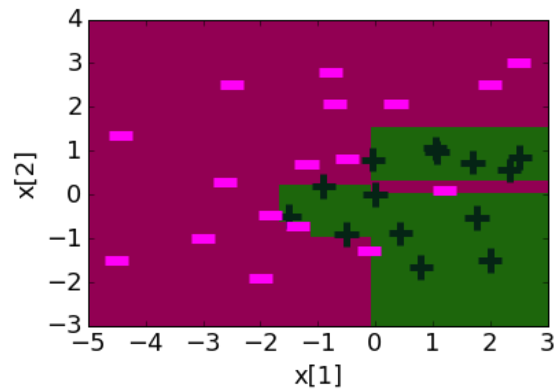
Depth 1



Depth 2

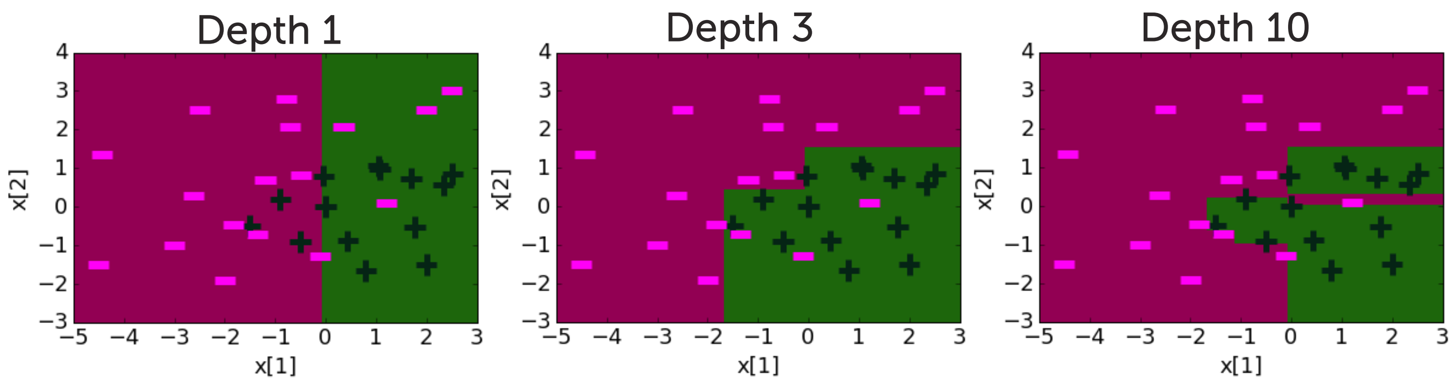


Depth 10

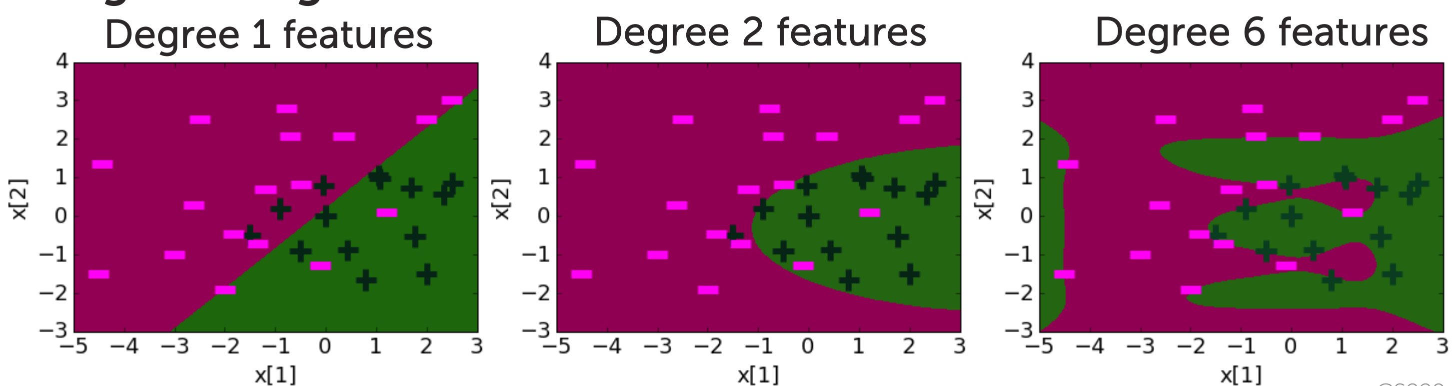


Comparing decision boundaries

Decision Tree



Logistic Regression



Summary of decision trees

What you can do now

- Define a decision tree classifier
- Interpret the output of a decision trees
- Learn a decision tree classifier using greedy algorithm
- Traverse a decision tree to make predictions
  - Majority class predictions
- Tackle continuous and discrete features

Annotated decisiontrees Slides

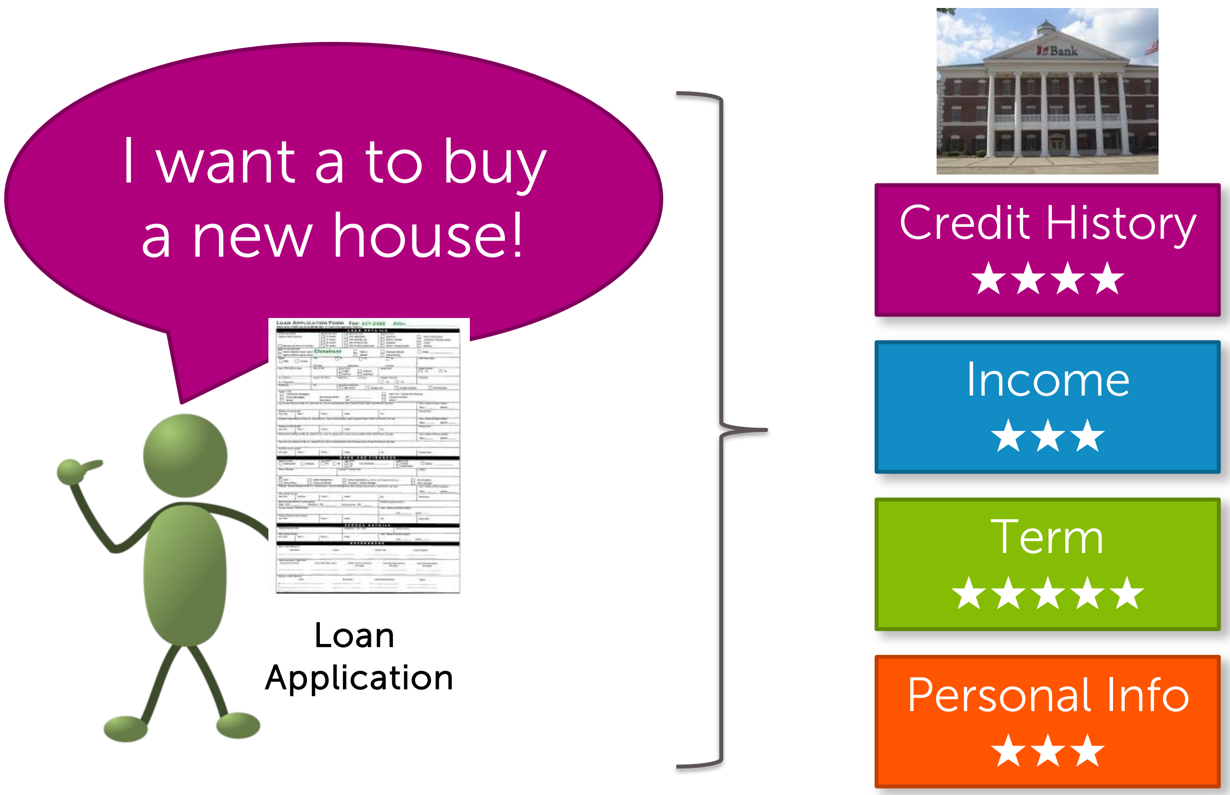
Decision Trees

CS229: Machine Learning
Carlos Guestrin
Stanford University

Slides include content developed by and co-developed with Emily Fox

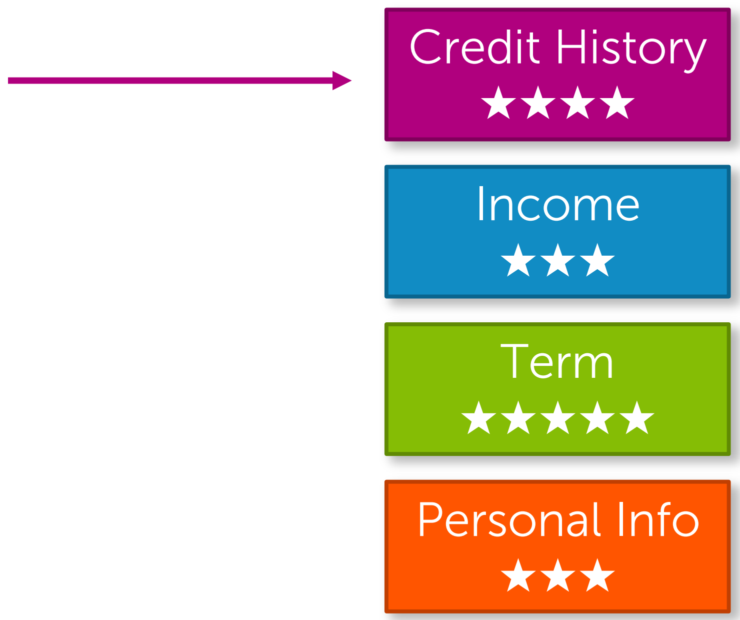
Predicting potential loan defaults

What makes a loan risky?



Credit history explained

Did I pay previous loans on time?

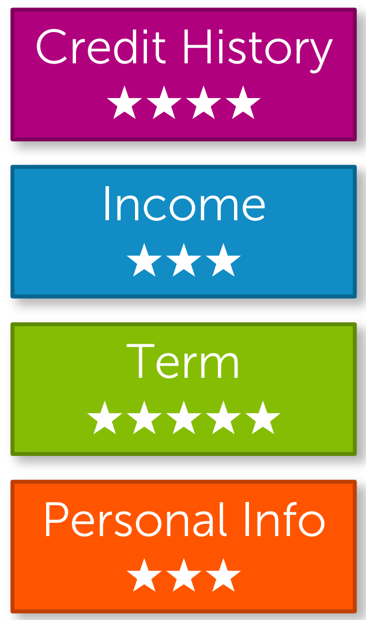


Example:
excellent, good, or fair

Income

What's my income?

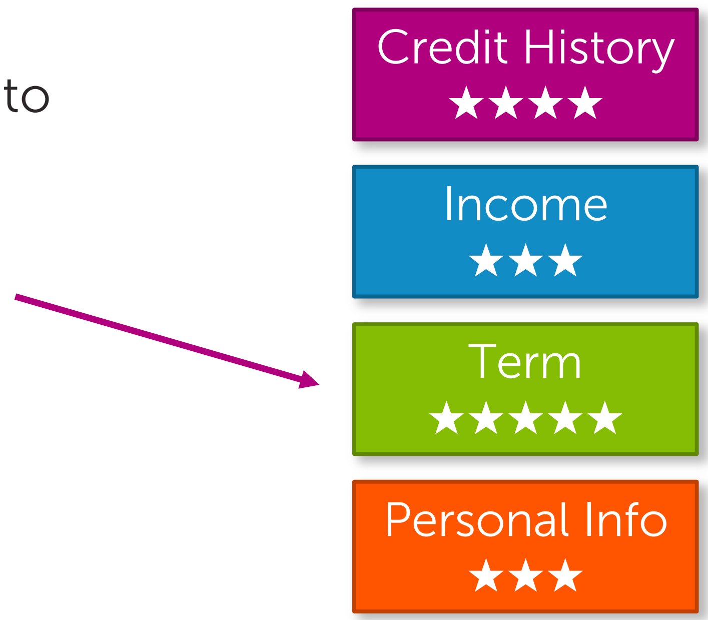
Example:
$80K per year



Loan terms

How soon do I need to pay the loan?

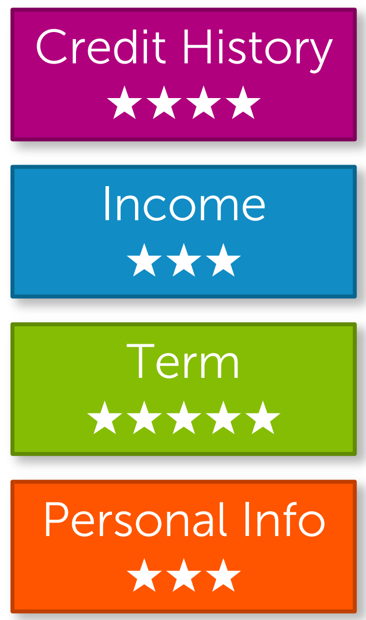
Example: 3 years, 5 years,…



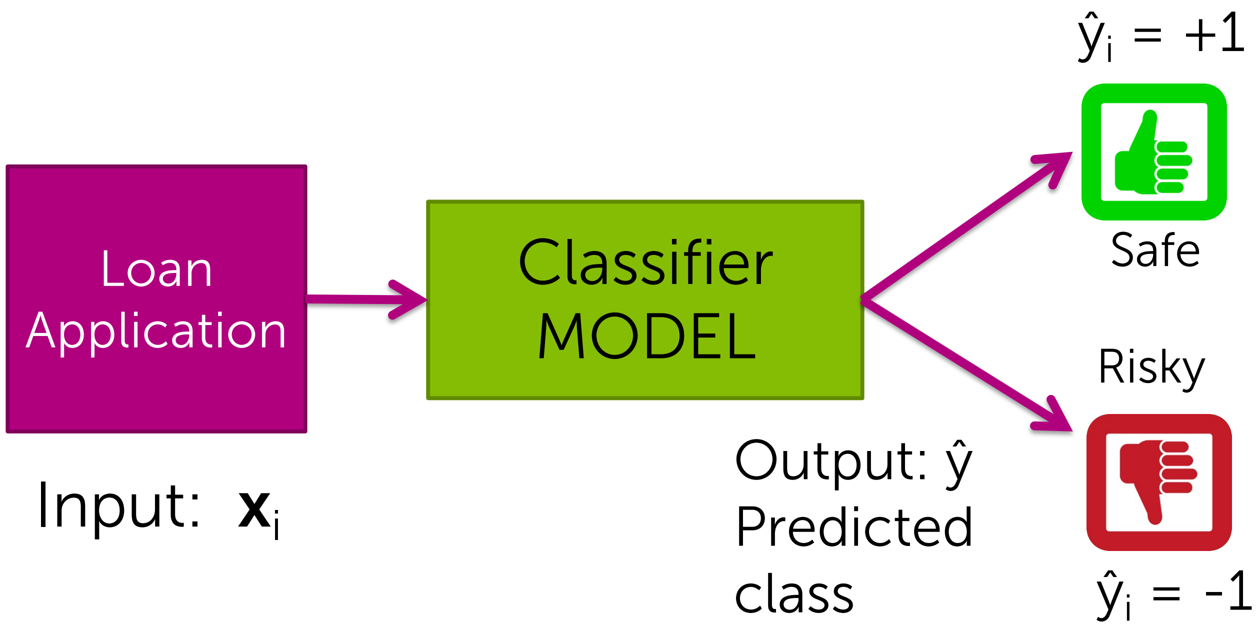
Personal information

Age, reason for the loan, marital status,…

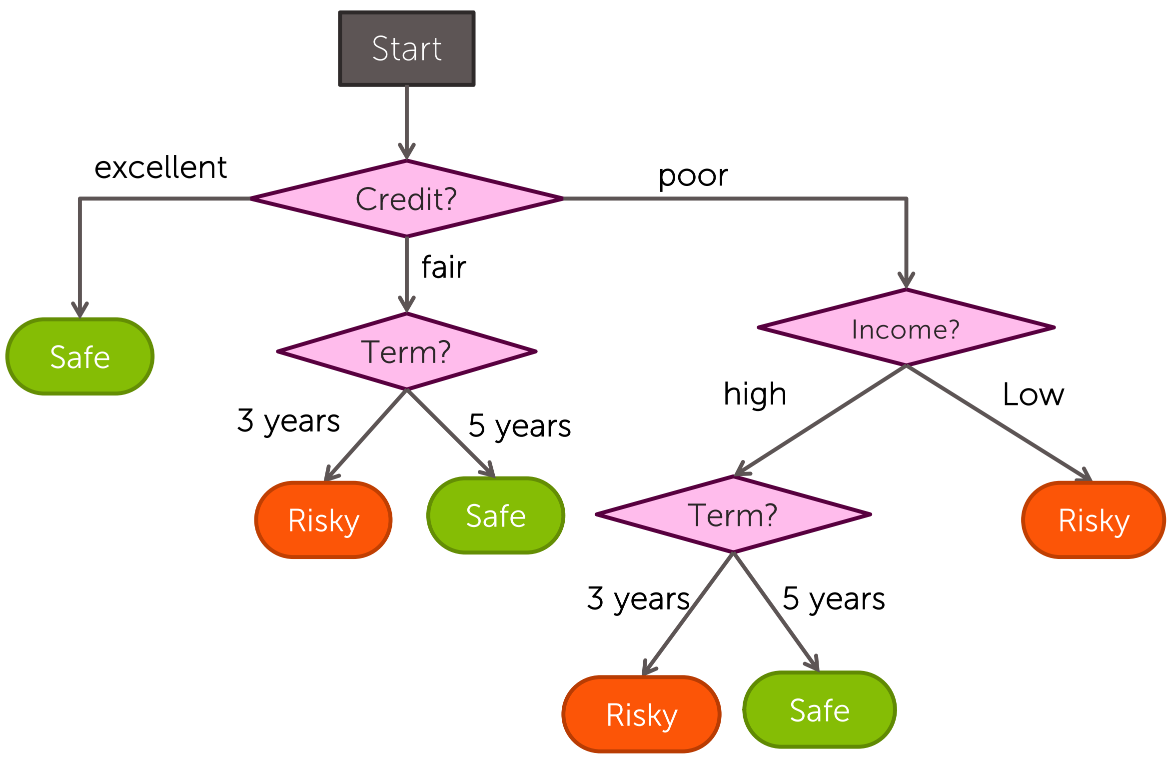
Example: Home loan for a married couple



Classifier review

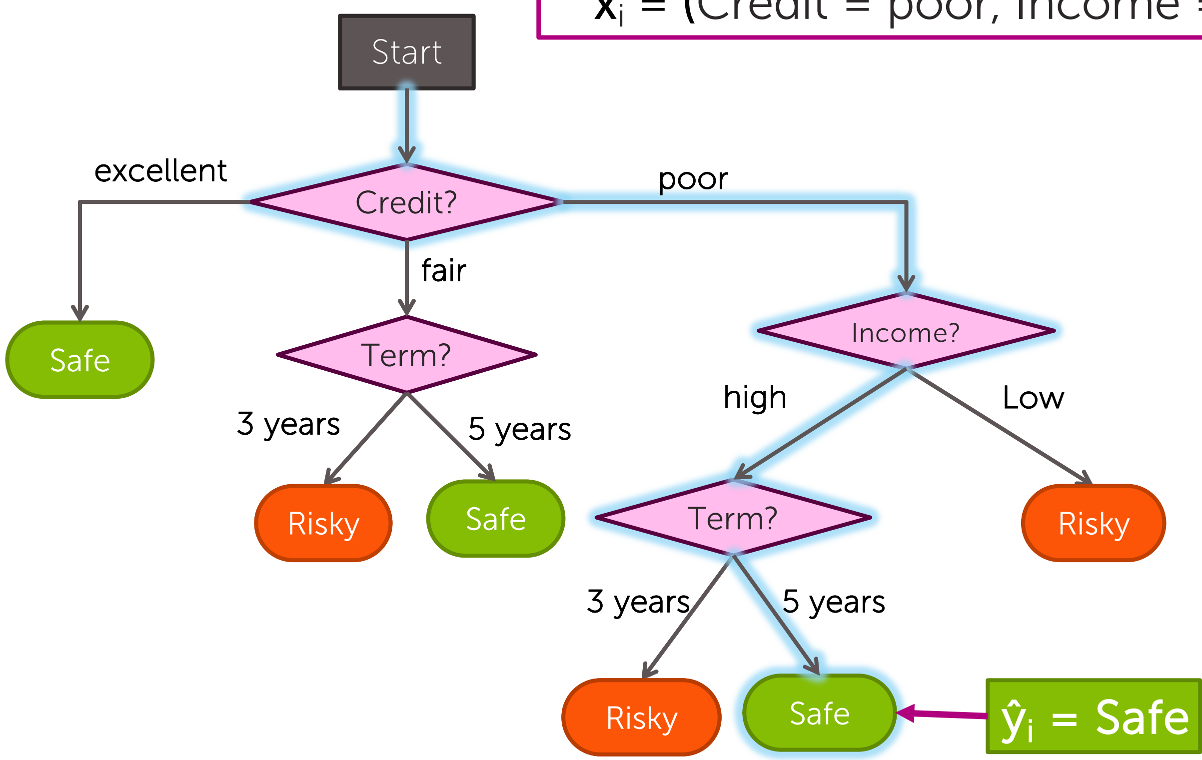


This module … decision trees



Scoring a loan application

$\mathbf{x}_i$ = (Credit = poor, Income = high, Term = 5 years)

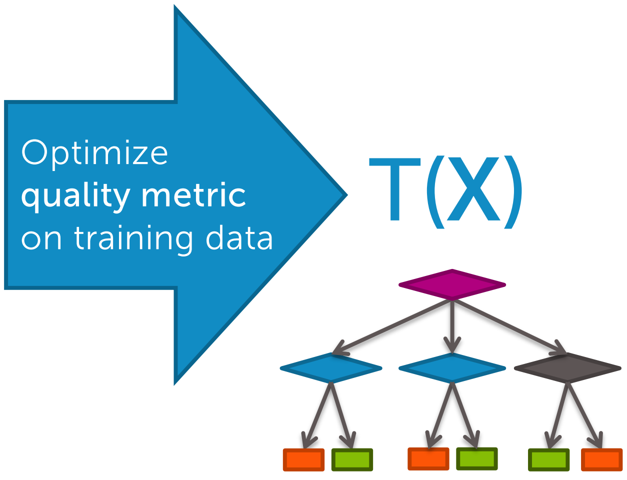


Decision tree learning task

Decision tree learning problem

Training data: N observations ($\mathbf{x}_i$,y_i)

Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe



Quality metric: Classification error

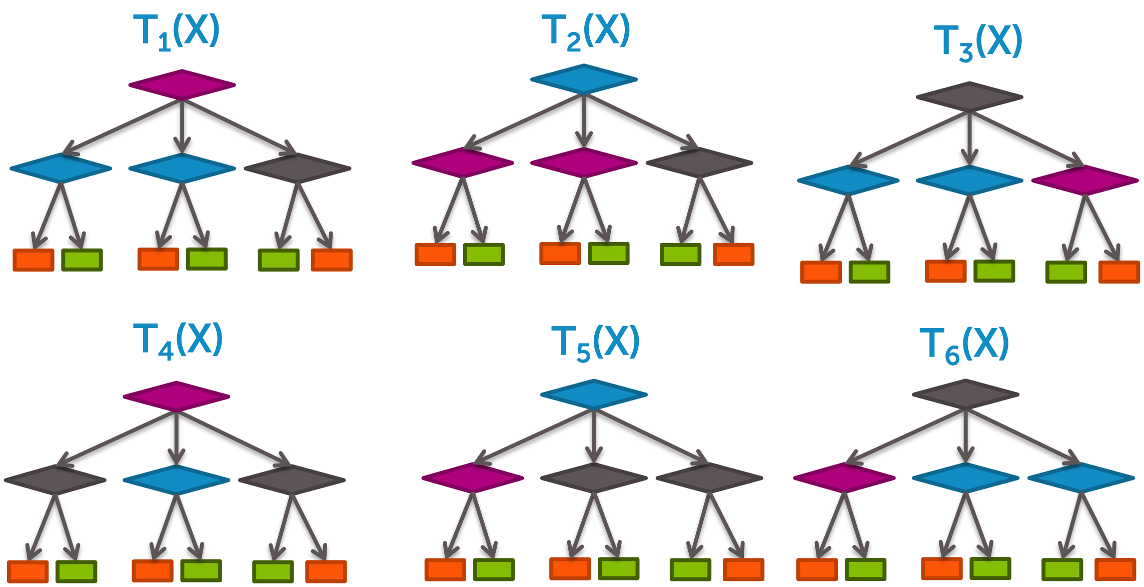
- Error measures fraction of mistakes

$$\text{Error} = \frac{\#\ \text{incorrect predictions}}{\#\ \text{examples}}$$

  - Best possible value : 0.0
  - Worst possible value: 1.0

How do we find the best tree?

Exponentially large number of possible trees makes decision tree learning hard!



Learning the smallest decision tree is an *NP-hard problem* [Hyafil & Rivest '76]

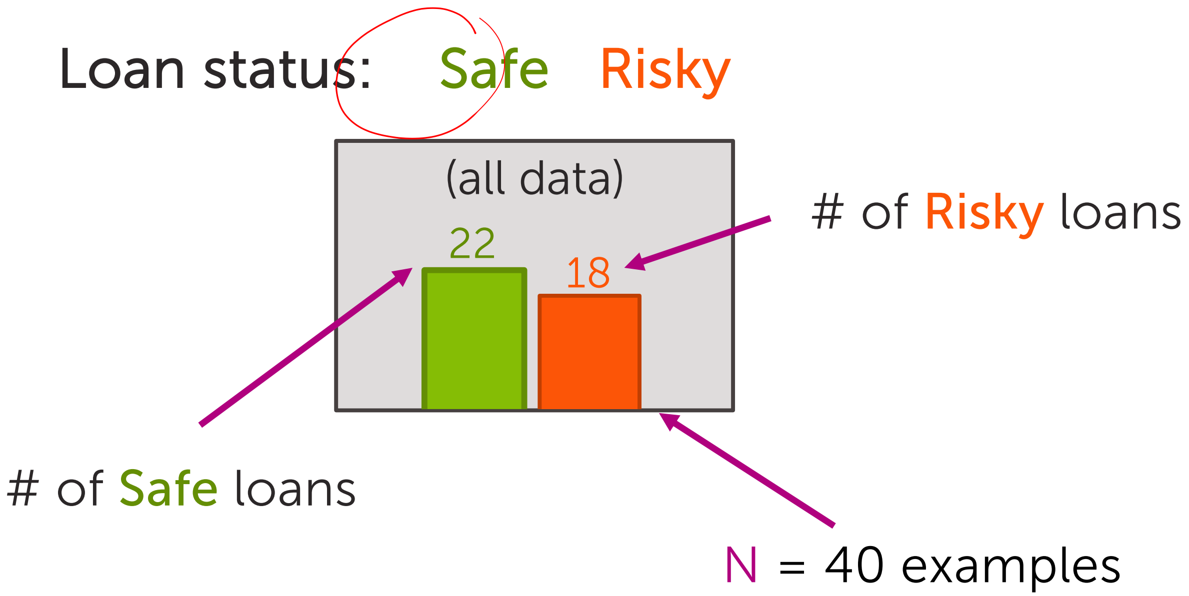
Greedy decision tree learning

Our training data table

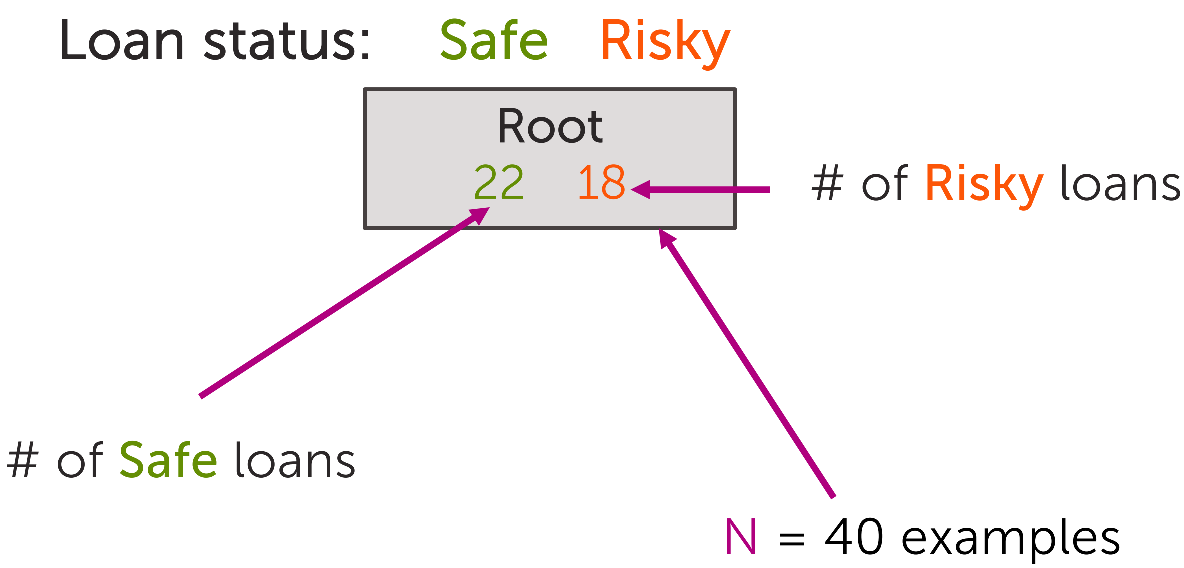
Assume N = 40, 3 features

Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe

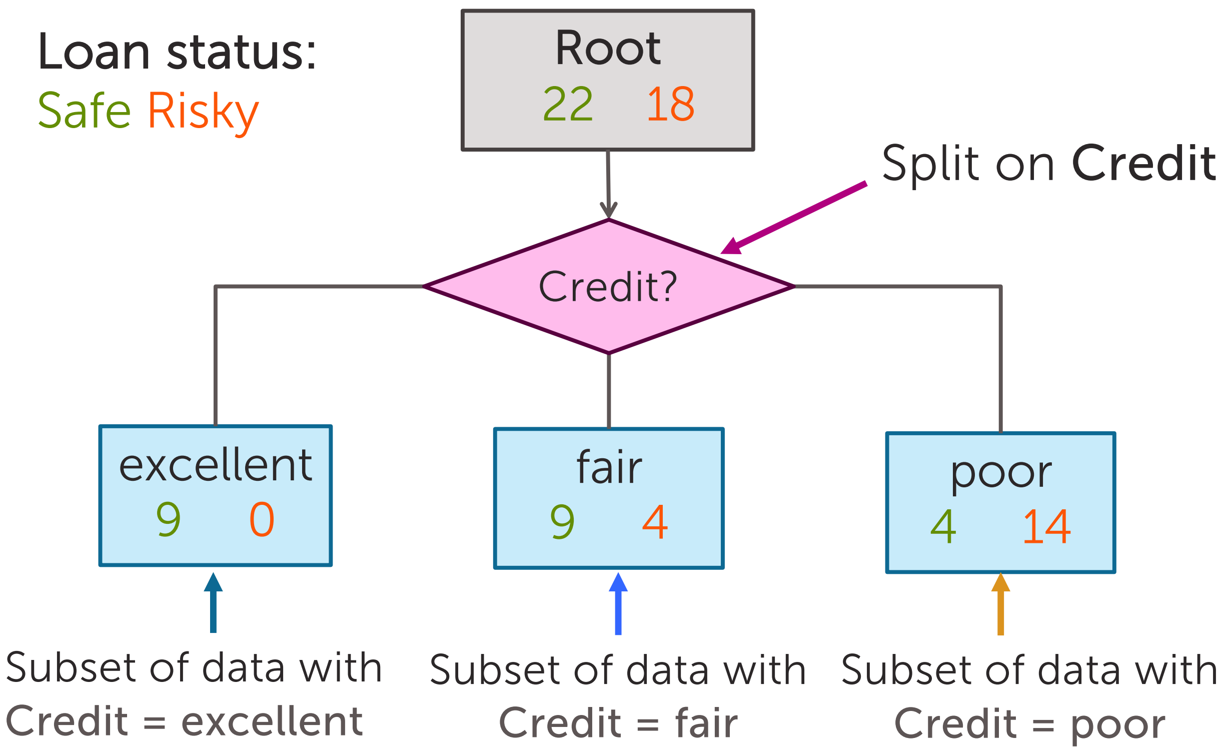
Start with all the data



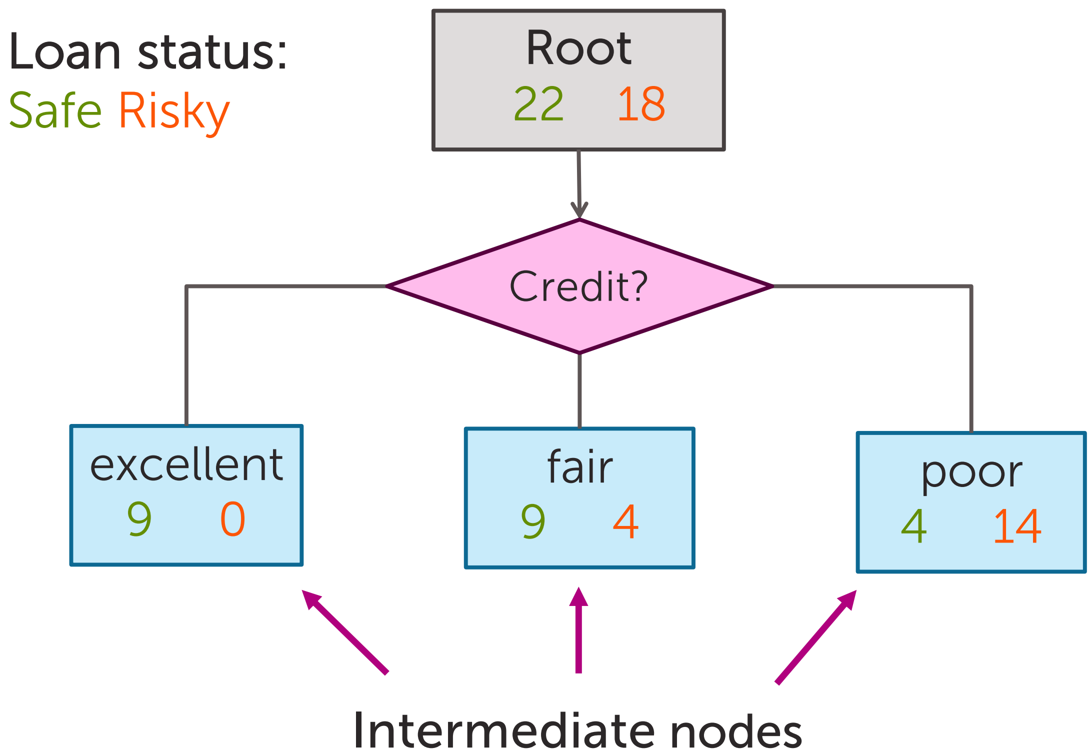
Compact visual notation: Root node



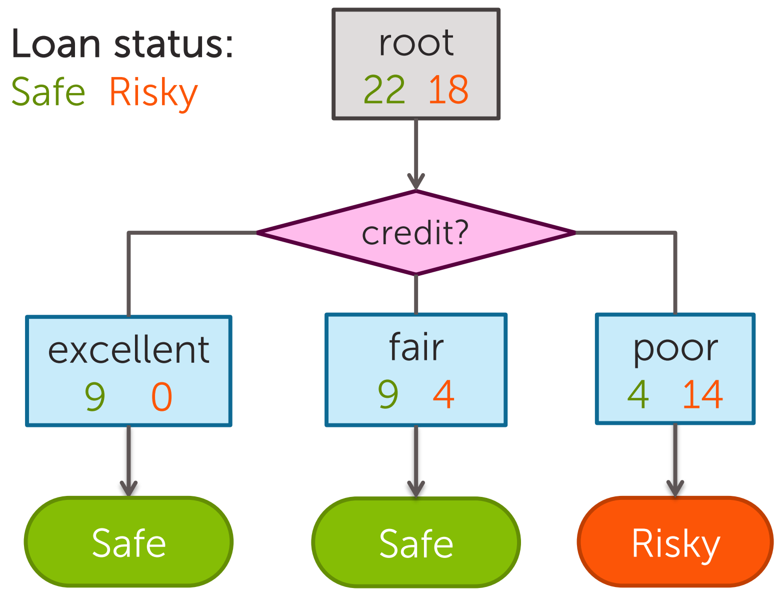
Decision stump: Single level tree



Visual notation: Intermediate nodes



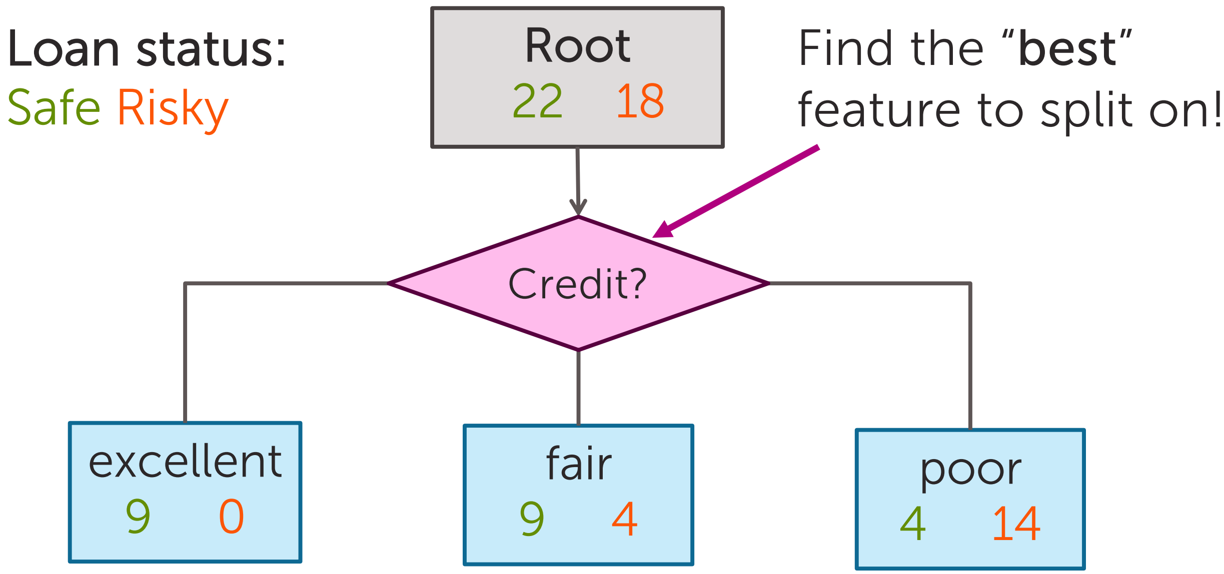
Making predictions with a decision stump



For each intermediate node, set $\hat{y}$ = **majority value**

Selecting best feature to split on

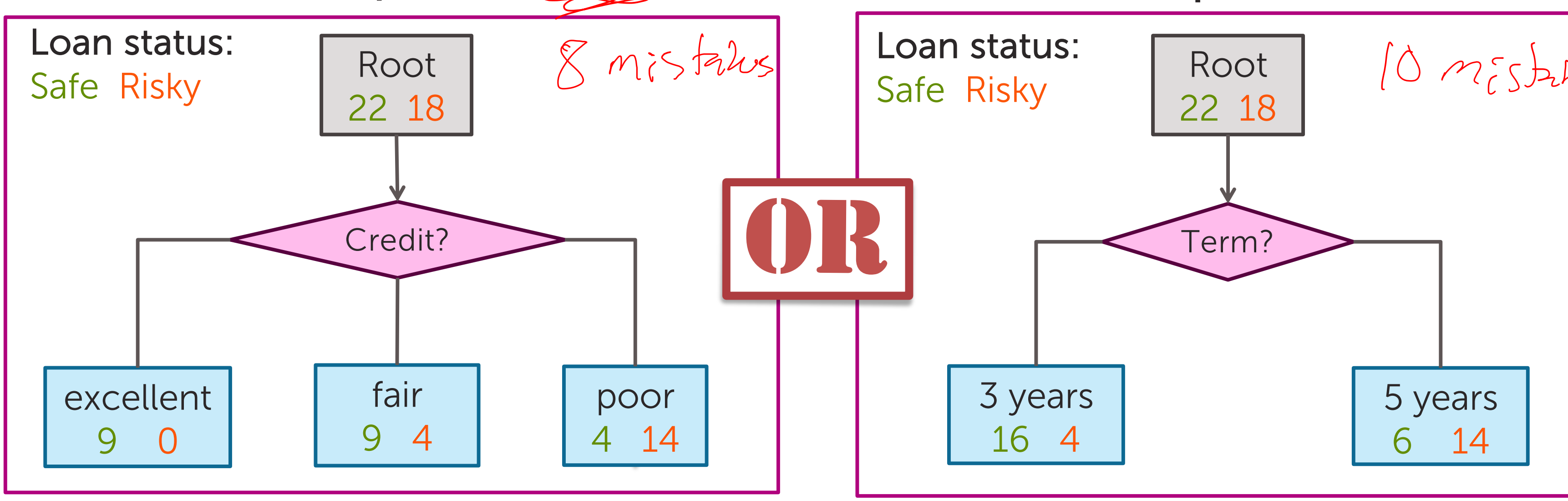
How do we learn a decision stump?



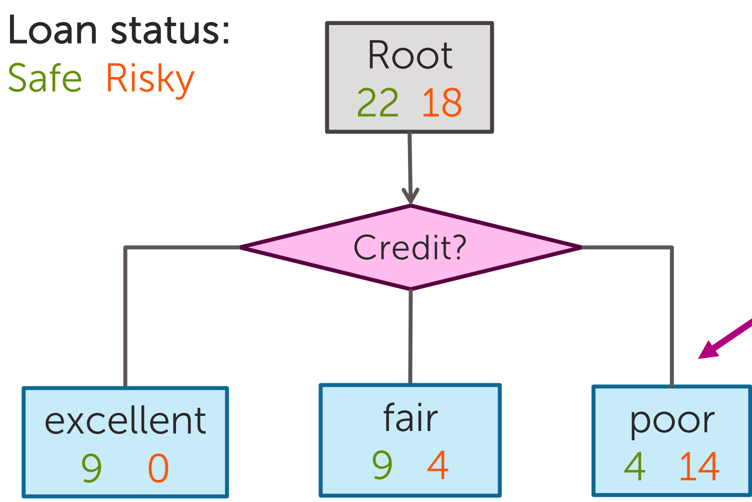
How do we select the best feature?

which one makes fewer mistakes?

Choice 1: Split on **Credit** | **Choice 2:** Split on **Term**



How do we measure effectiveness of a split?

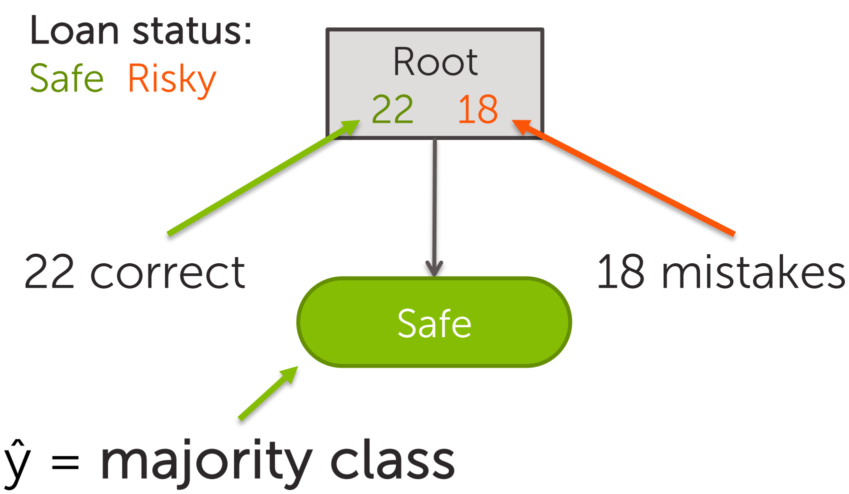


Idea: Calculate classification error of this decision stump

$$\text{Error} = \frac{\#\ \text{mistakes}}{\#\ \text{data points}}$$

Calculating classification error

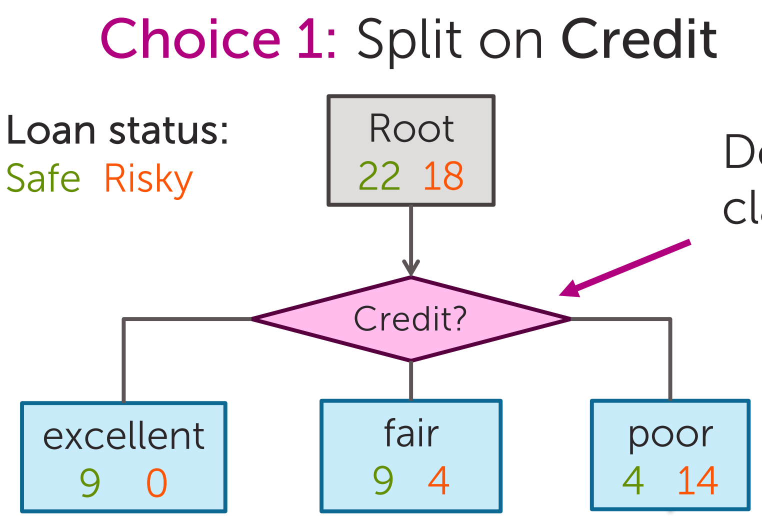
- **Step 1:** ŷ = class of majority of data in node
- **Step 2:** Calculate classification error of predicting ŷ for this data



$$\text{Error} = \frac{18}{40} = 0.45$$

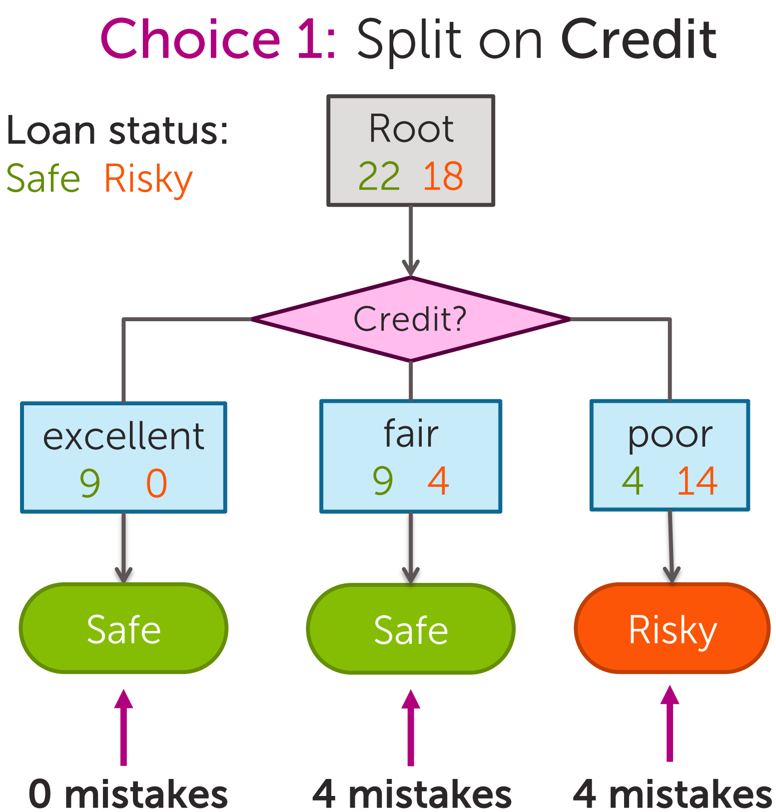
Tree	Classification error
(root)	0.45

Choice 1: Split on Credit history?



Does a split on Credit reduce classification error below 0.45?

Split on Credit: Classification error

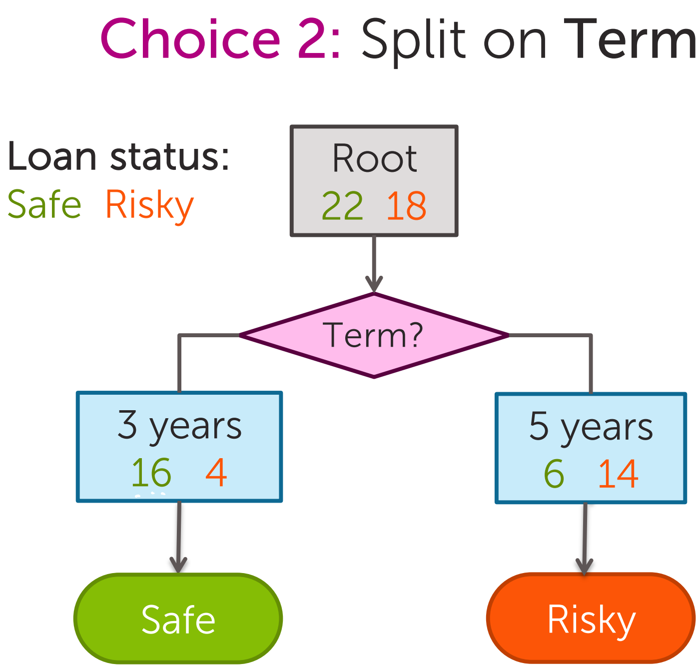


Error = $\frac{8}{40}$

= 0.2

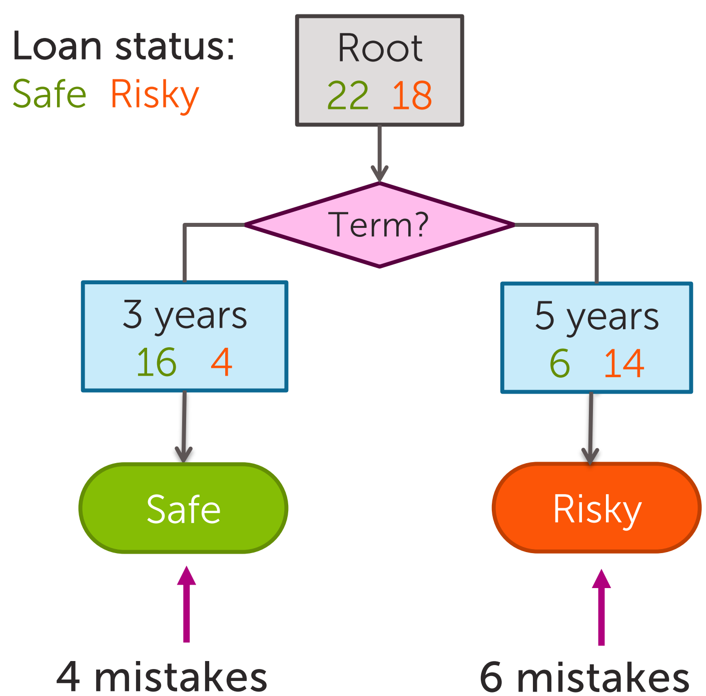
Tree	Classification error
(root)	0.45
Split on **credit**	0.2

Choice 2: Split on Term?



Evaluating the split on Term

Choice 2: Split on **Term**



Error = $\frac{10}{40}$

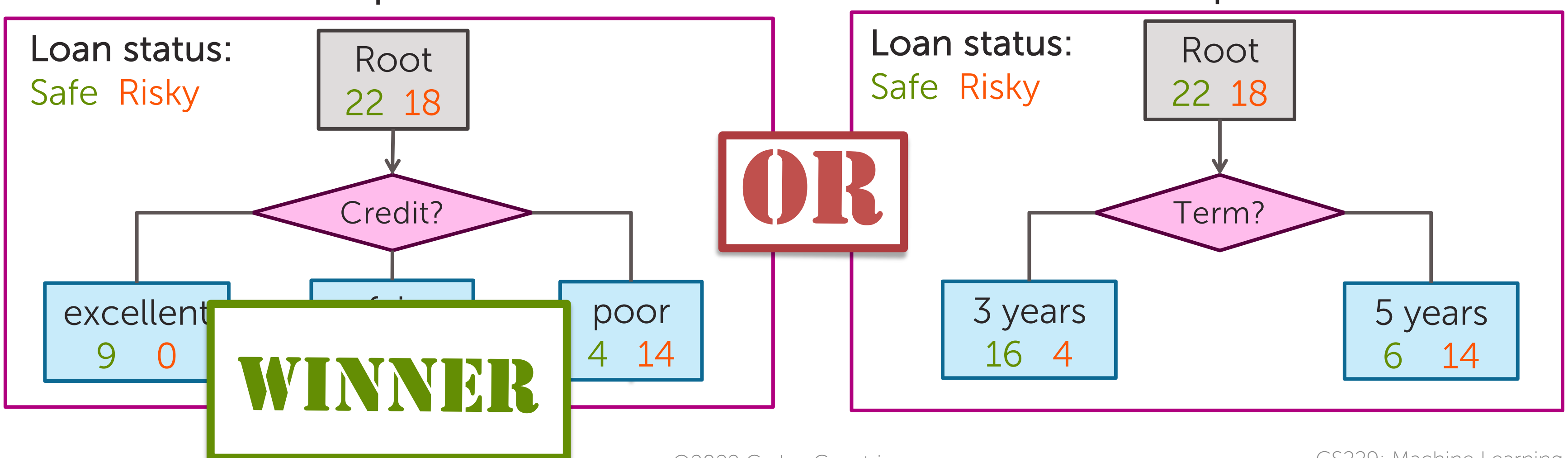
= 0.25

Tree	Classification error
(root)	0.45
Split on **credit**	0.2
Split on **term**	0.25

Choice 1 vs Choice 2: Comparing split on Credit vs Term

Tree	Classification error
(root)	0.45
split on **credit**	0.2
split on **loan term**	0.25

Choice 1: Split on **Credit** **Choice 2:** Split on **Term**

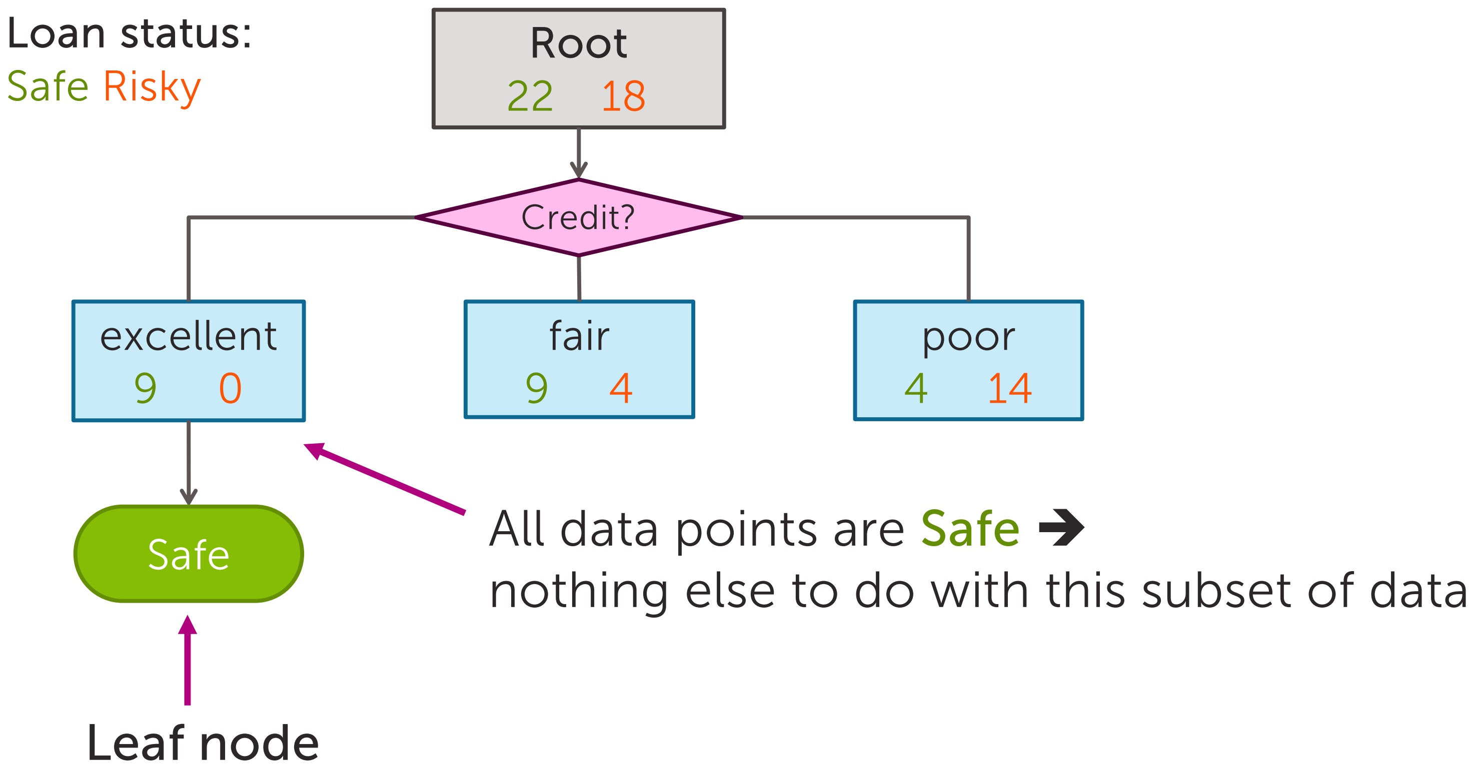


Feature split selection algorithm

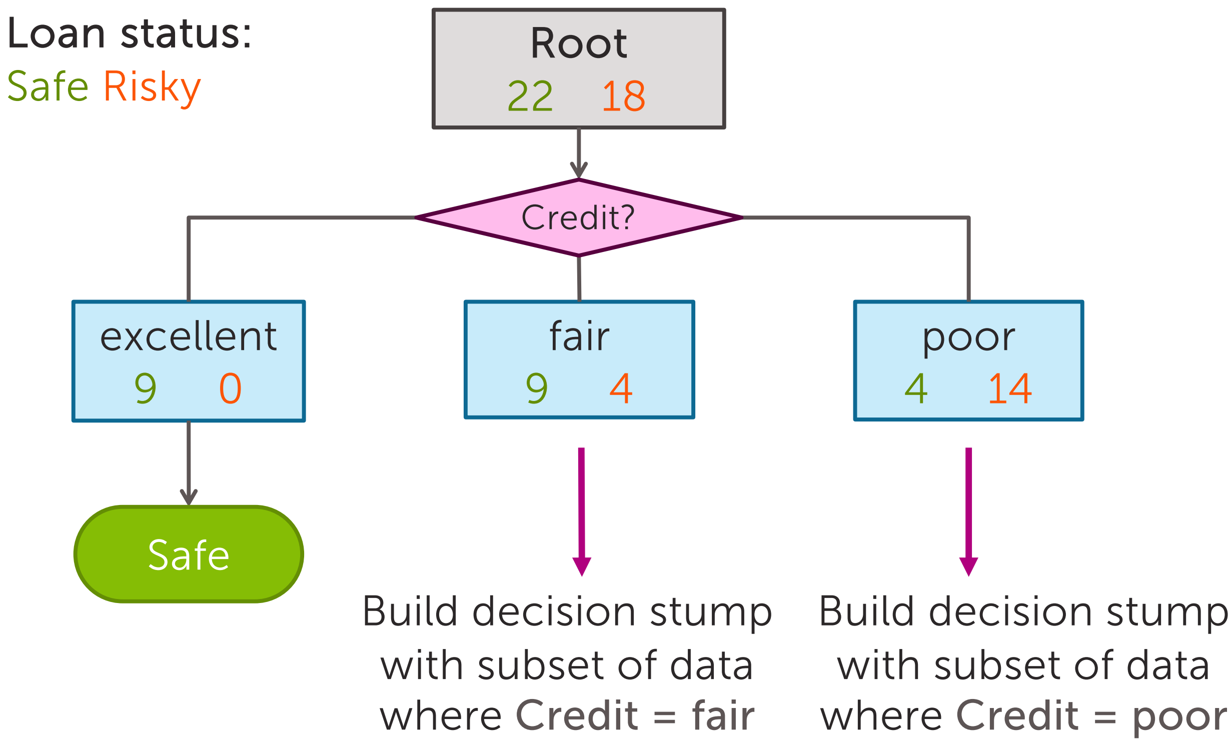
- Given a subset of data M (a node in a tree)
- For each feature $h_i(x)$:
  1. Split data of M according to feature $h_i(x)$
  2. Compute classification error of split
- Chose feature $h^*(x)$ with lowest classification error

Recursion & Stopping conditions

We've learned a decision stump, what next?

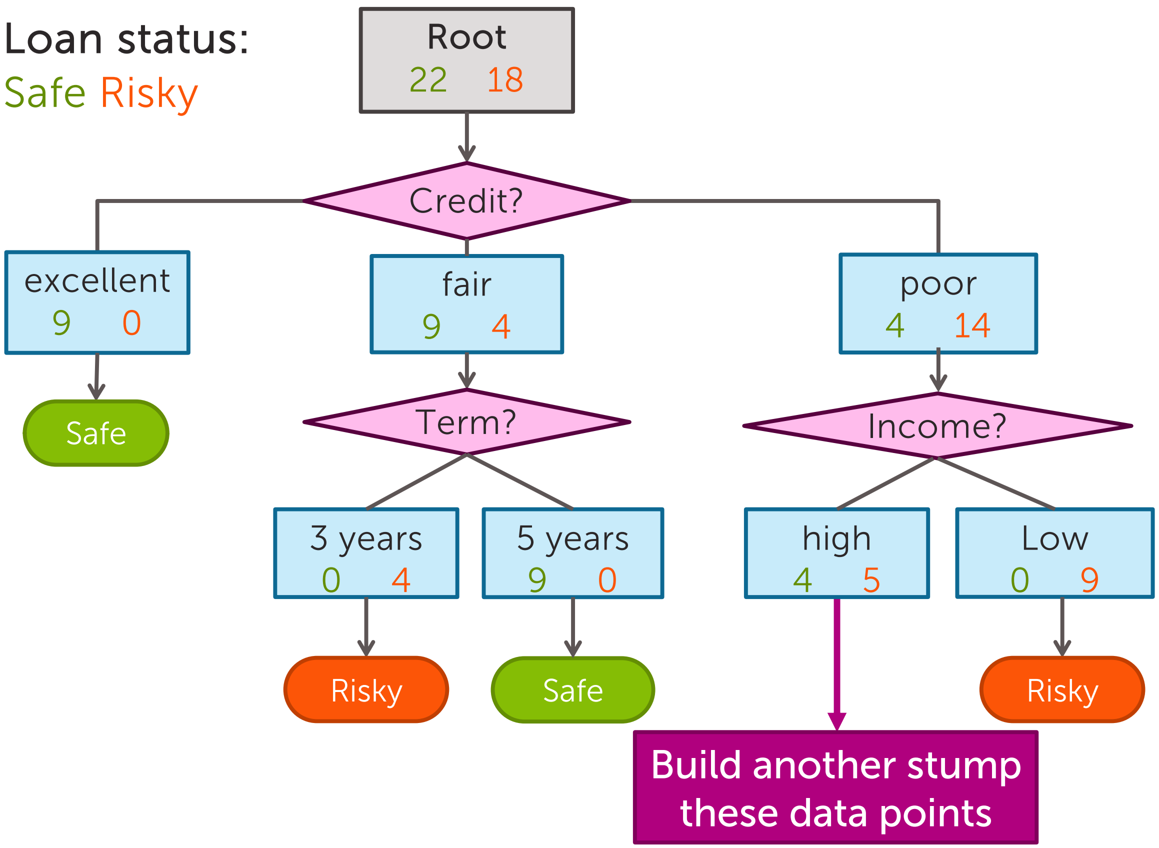


Tree learning = Recursive stump learning

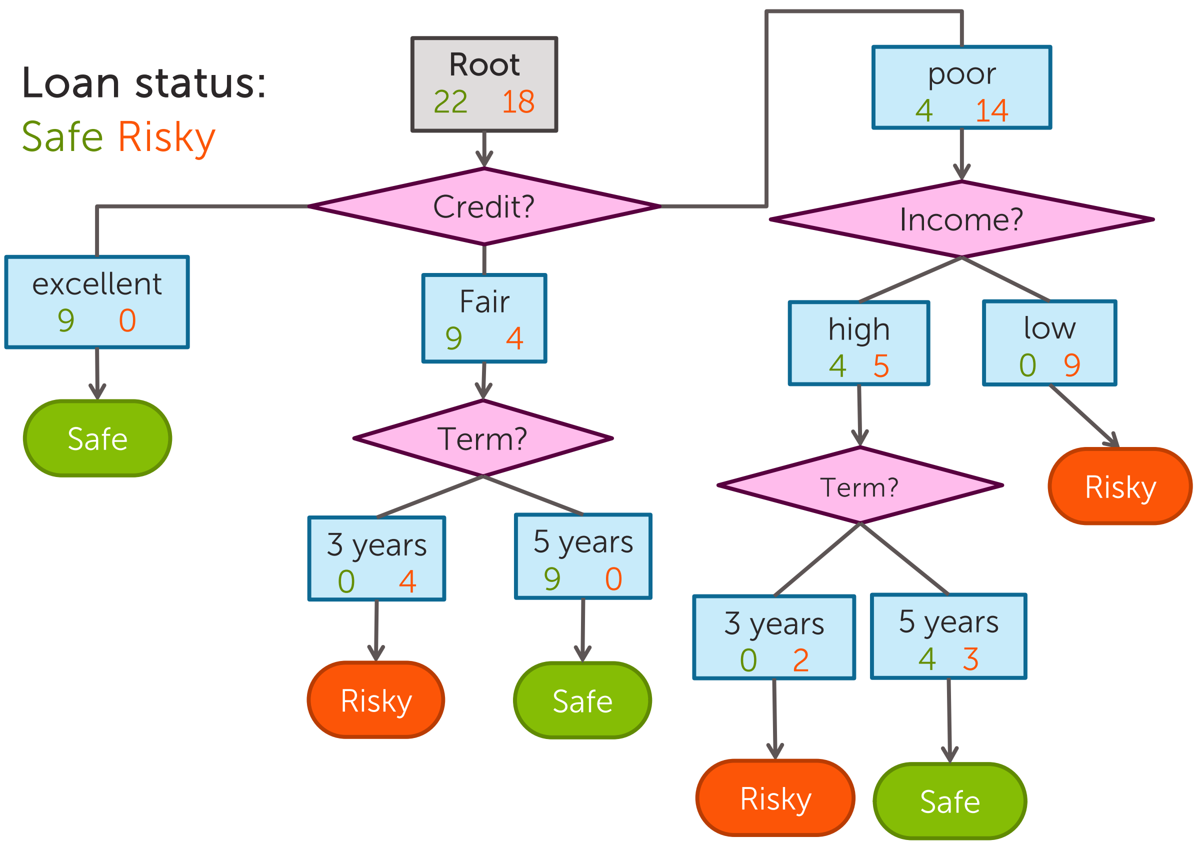


Second level

Loan status:
Safe Risky



Final decision tree



Simple greedy decision tree learning

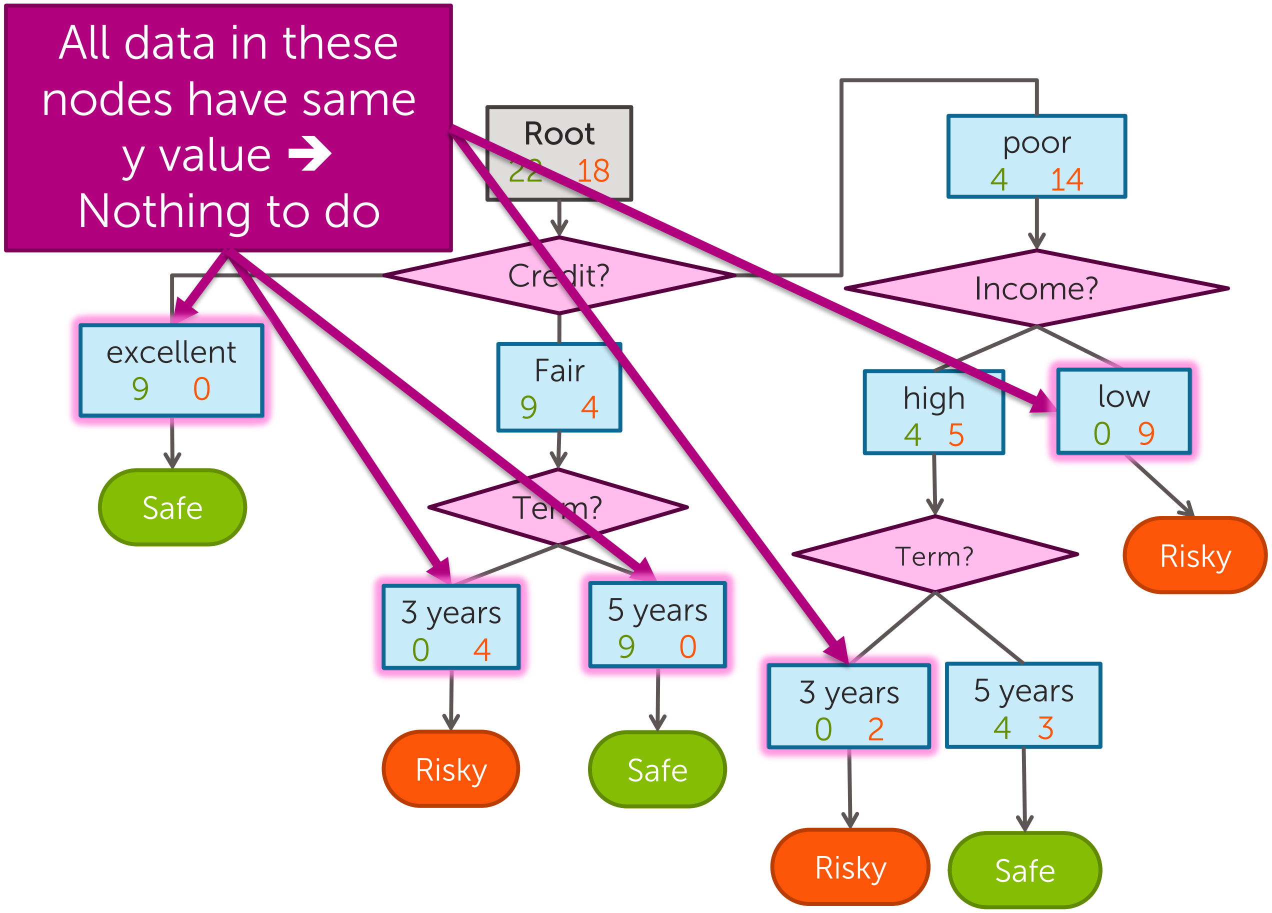
Pick best feature to split on

Learn decision stump with this split

For each leaf of decision stump, recurse

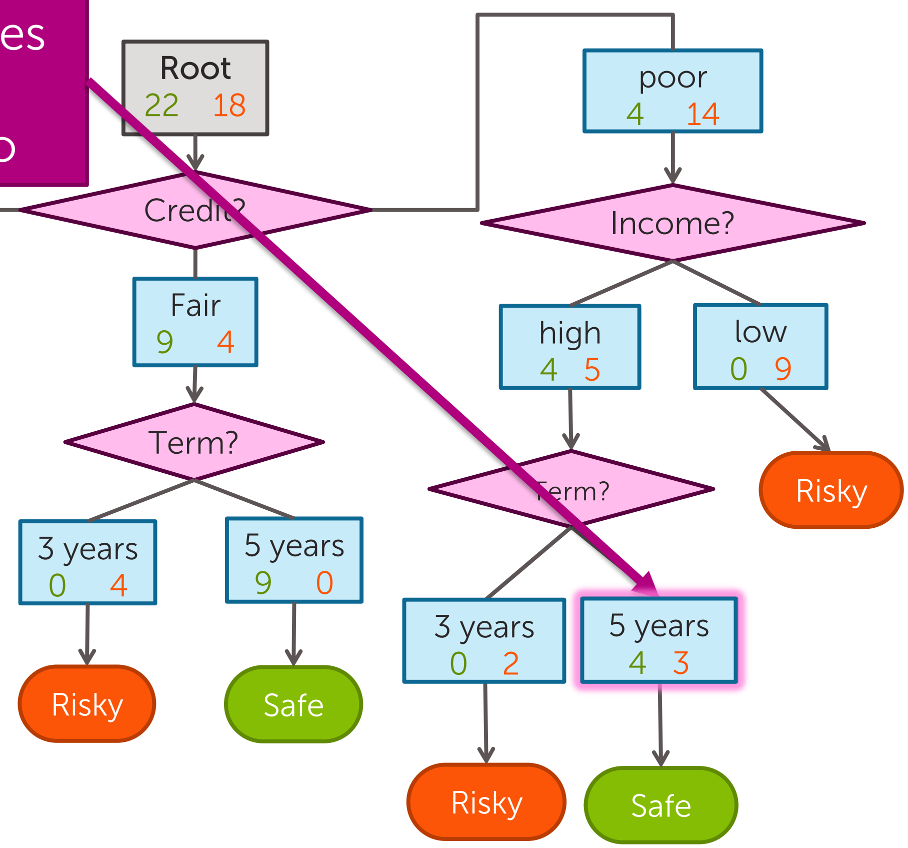
When do we stop???

Stopping condition 1: All data agrees on y



Stopping condition 2: Already split on all features

Already split on all possible features
➔
Nothing to do



Greedy decision tree learning

- Step 1: Start with an empty tree
- Step 2: Select a feature to split data
- For each split of the tree:
  - Step 3: If nothing more to do, make predictions
  - Step 4: Otherwise, go to Step 2 & continue (recurse) on this split

Pick feature split leading to lowest classification error

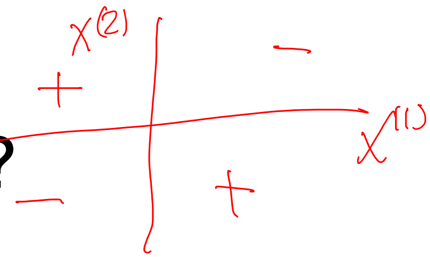
Stopping conditions

Recursion

Is this a good idea?

Proposed stopping condition 3:
Stop if no split reduces the classification error

Stopping condition 3:
Don't stop if error doesn't decrease???



y = **x**[1] xor **x**[2]

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False

y values
True False

Root	
2	2

Error = $\frac{2}{4}$

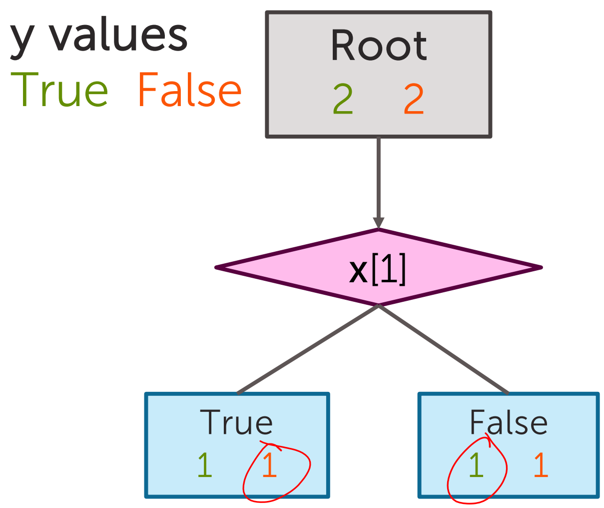
= 0.5

Tree	Classification error
(root)	0.5

Consider split on x[1]

y = **x**[1] xor **x**[2]

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False



Error = $\frac{2}{4}$

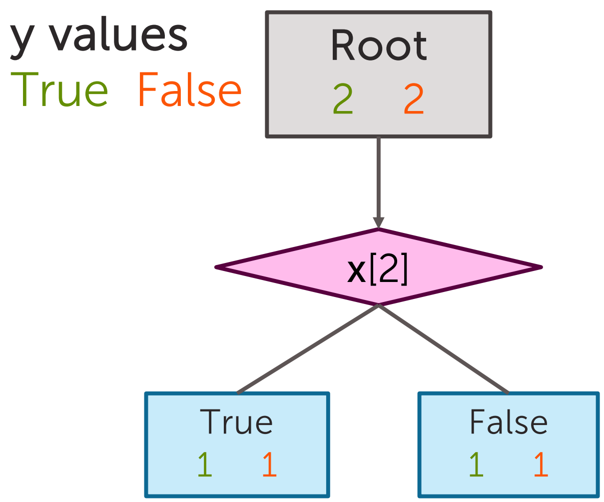
= 0.5

Tree	Classification error
(root)	0.5
Split on **x**[1]	0.5

Consider split on x[2]

$y = \mathbf{x}[1]$ xor $\mathbf{x}[2]$

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False



$$\text{Error} = \frac{1+1}{2+2} = 0.5$$

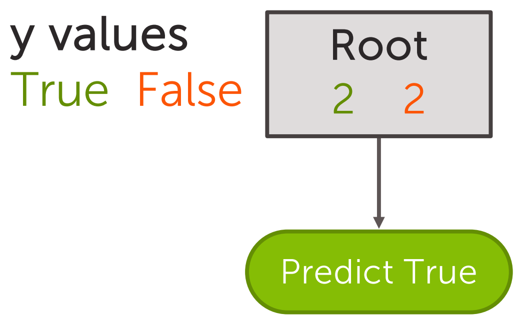
Neither features improve training error… Stop now???

Tree	Classification error
(root)	0.5
Split on **x**[1]	0.5
Split on **x**[2]	0.5

Final tree with stopping condition 3

y = **x**[1] xor **x**[2]

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False



Tree	Classification error
with stopping condition 3	0.5

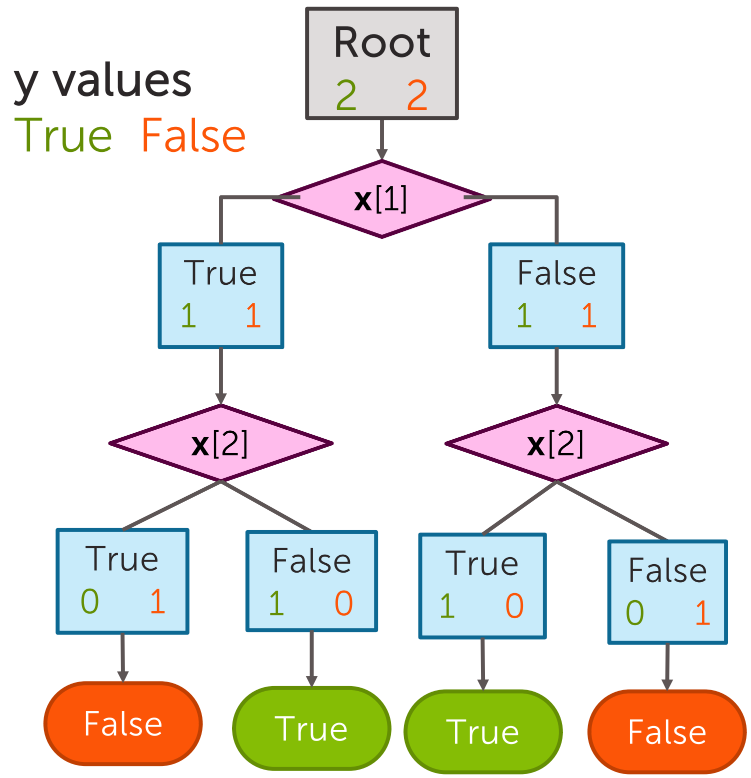
Without stopping condition 3

Condition 3 (stopping when training error doesn't' improve) is **not** recommended!

y = **x**[1] xor **x**[2]

x[1]	**x**[2]	y
False	False	False
False	True	True
True	False	True
True	True	False

Tree	Classification error
with stopping condition 3	0.5
without stopping condition 3	0

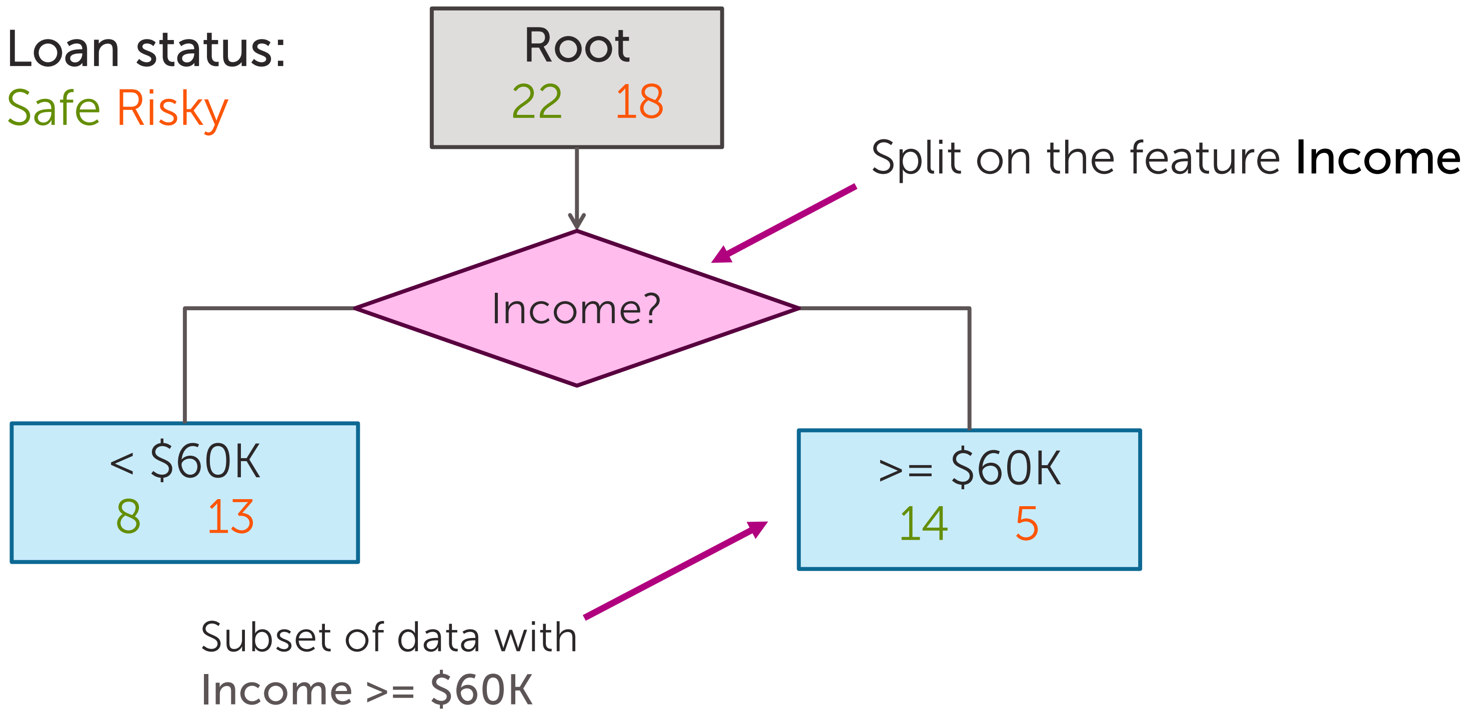


Decision tree learning:
Real valued features

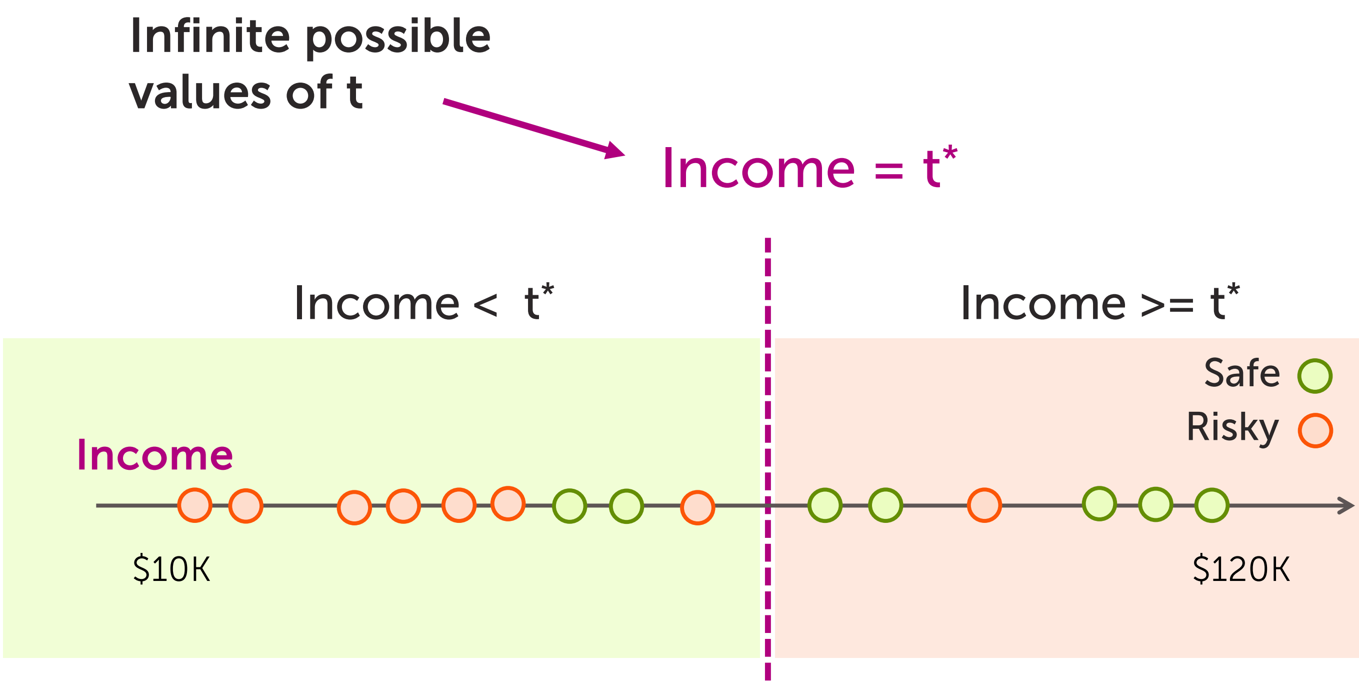
How do we use real values inputs?

Income	Credit	Term	y
$105 K	excellent	3 yrs	Safe
$112 K	good	5 yrs	Risky
$73 K	fair	3 yrs	Safe
$69 K	excellent	5 yrs	Safe
$217 K	excellent	3 yrs	Risky
$120 K	good	5 yrs	Safe
$64 K	fair	3 yrs	Risky
$340 K	excellent	5 yrs	Safe
$60 K	good	3 yrs	Risky

Threshold split → turn continuous var into binary

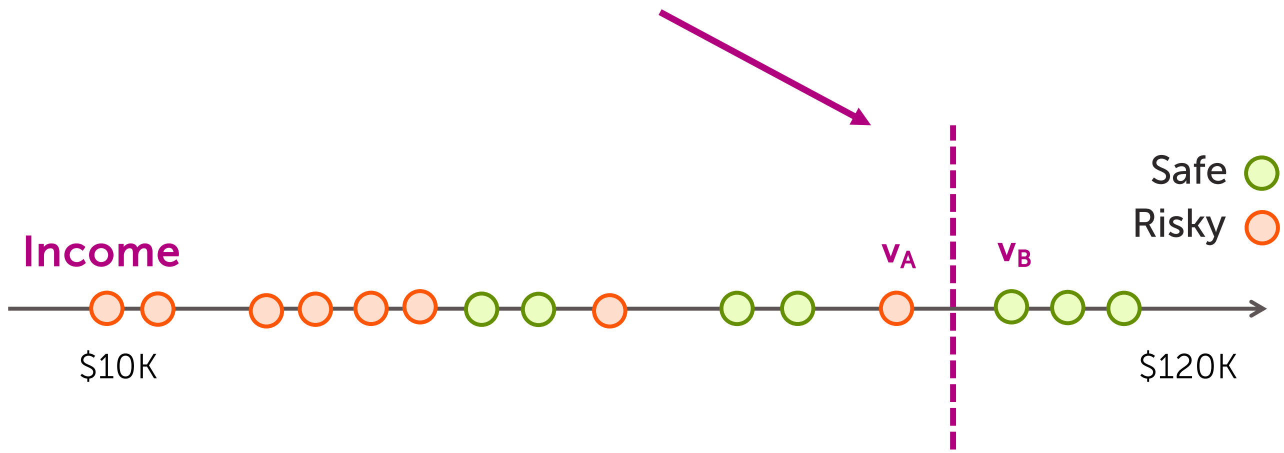


Finding the best threshold split



Consider a threshold between points

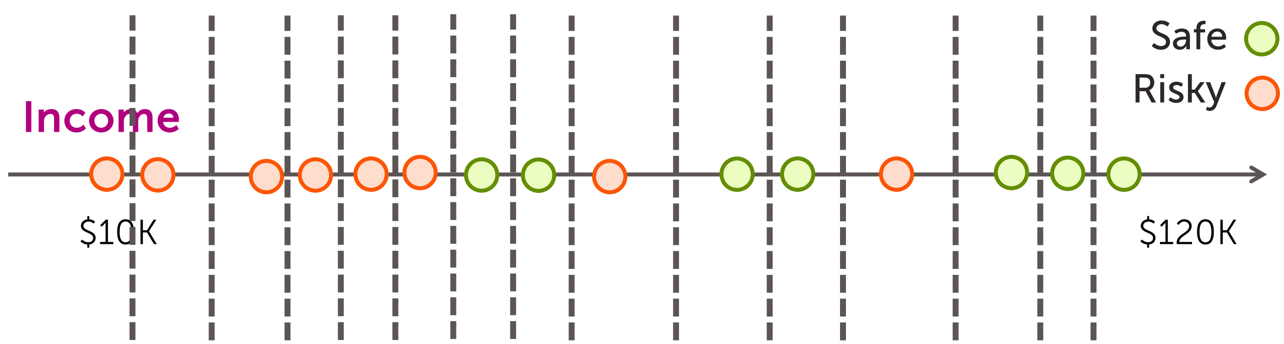
Same classification error for any threshold split between v_A and v_B



Only need to consider mid-points

Sort data:

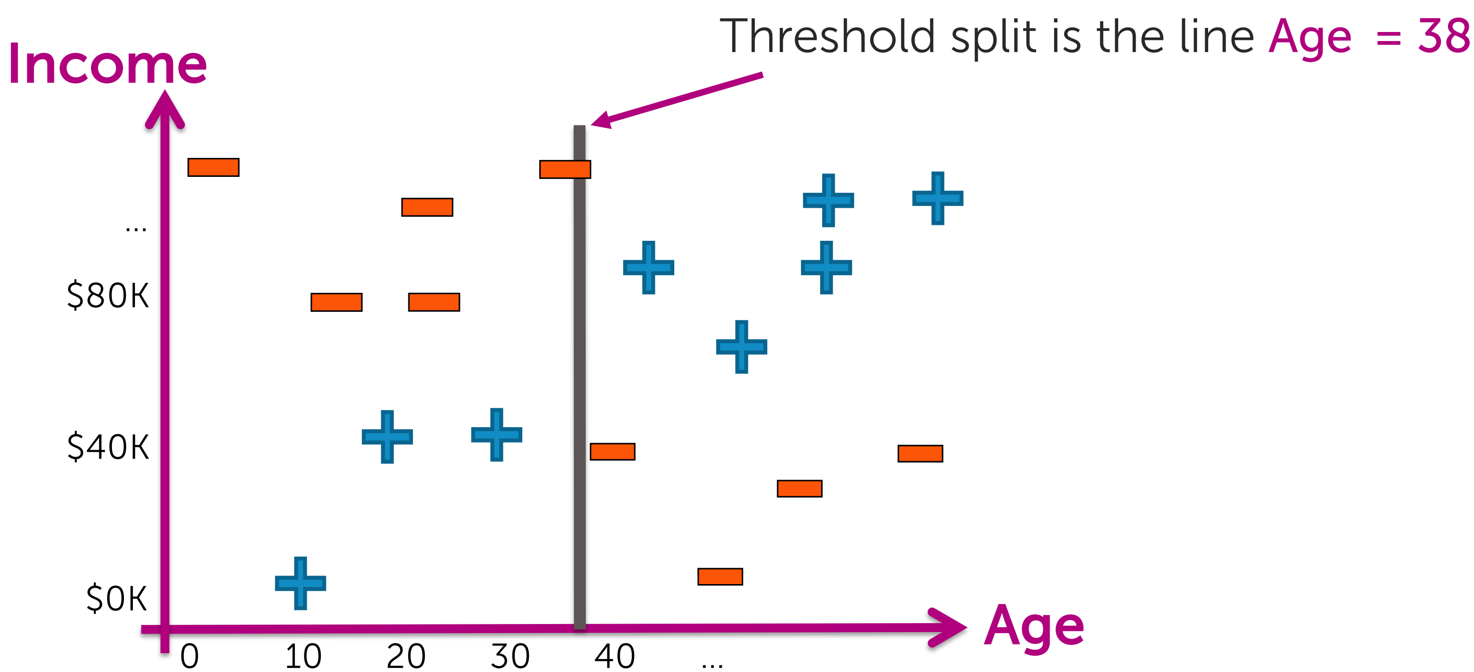
Finite number of
splits to consider



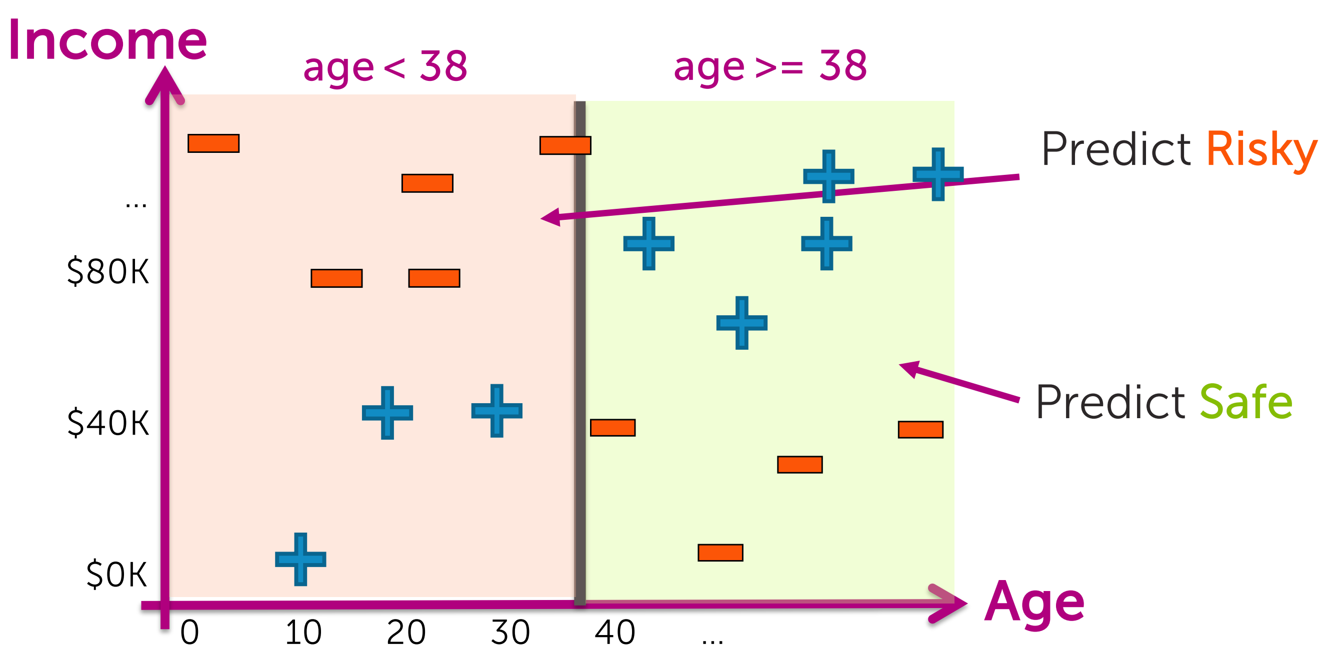
Threshold split selection algorithm

- Step 1: Sort the values of a feature $h_j(\mathbf{x})$:
  Let $\{v_1, v_2, v_3, \ldots v_N\}$ denote sorted values
- Step 2:
  - For i = 1 … N-1
    - Consider split $t_i = (v_i + v_{i+1}) / 2$
    - Compute classification error for treshold split $h_j(\mathbf{x}) >= \mathbf{t}_i$
  - Chose the $\mathbf{t}^*$ with the lowest classification error

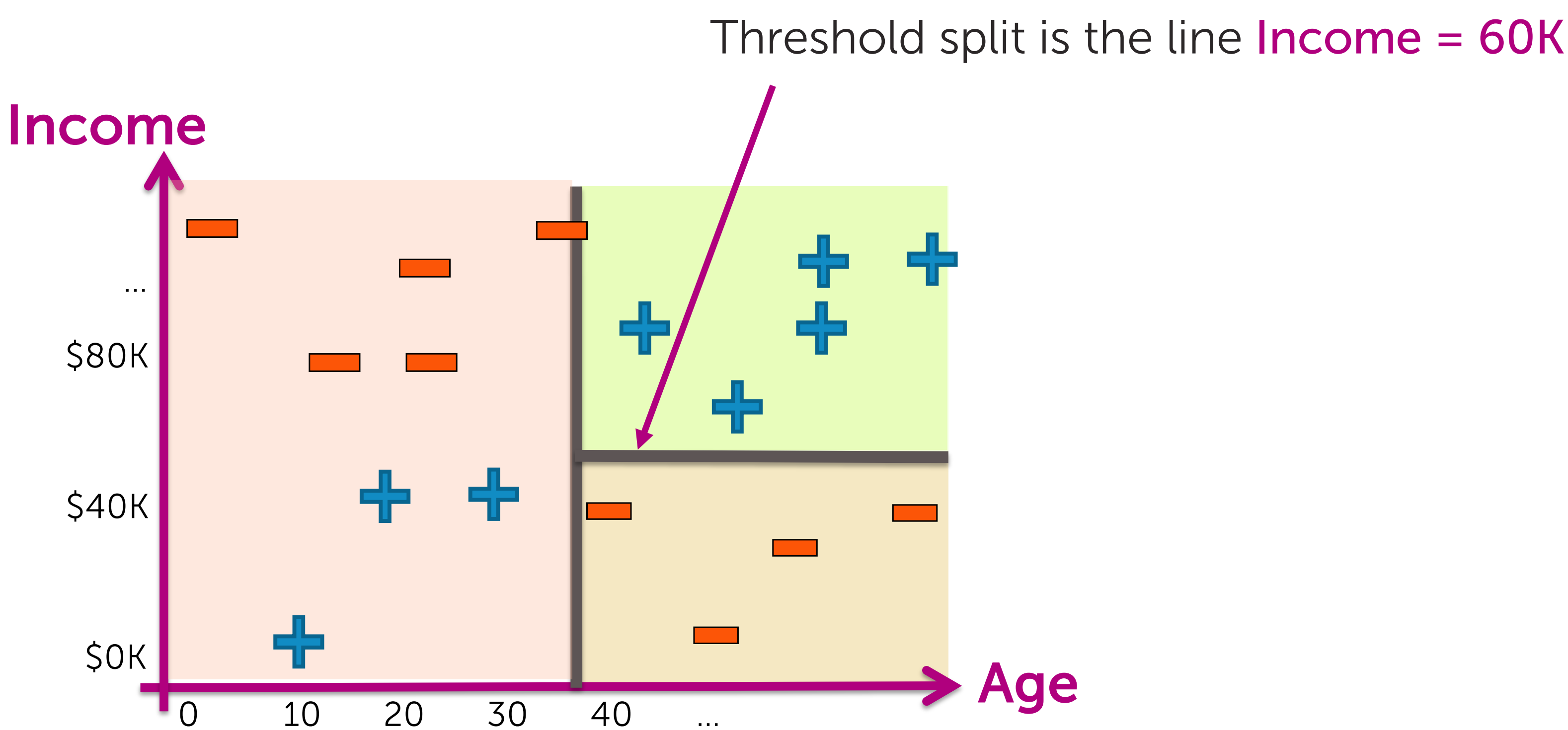
Visualizing the threshold split



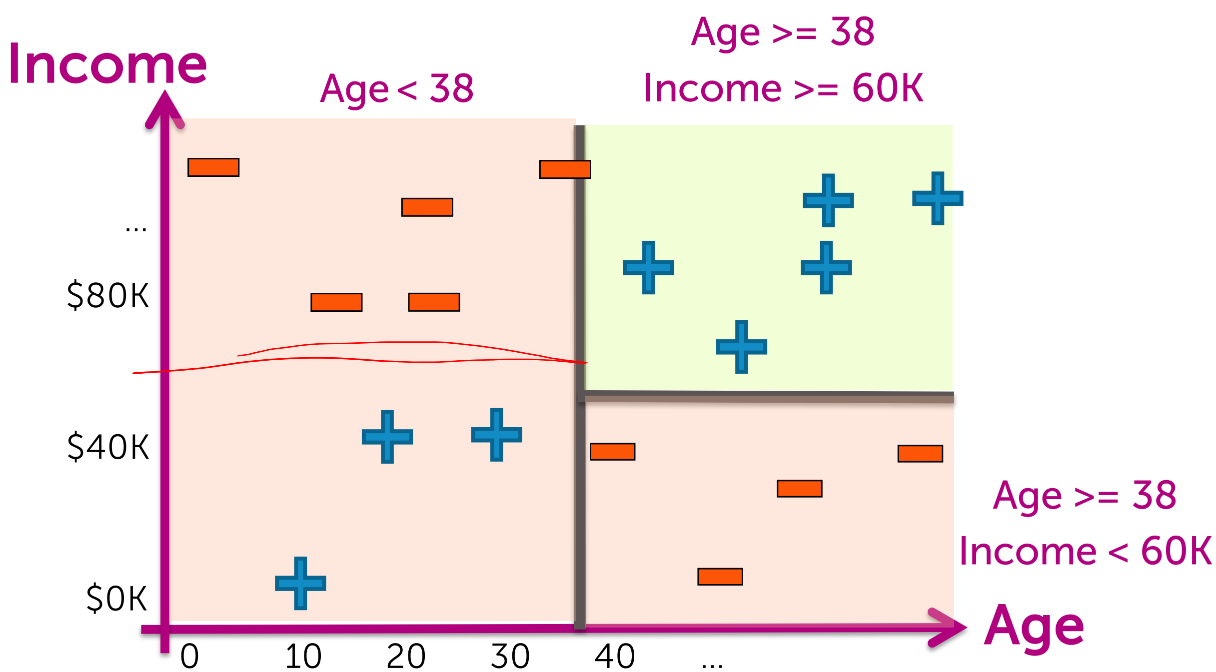
Split on Age >= 38



Depth 2: Split on Income >= $60K



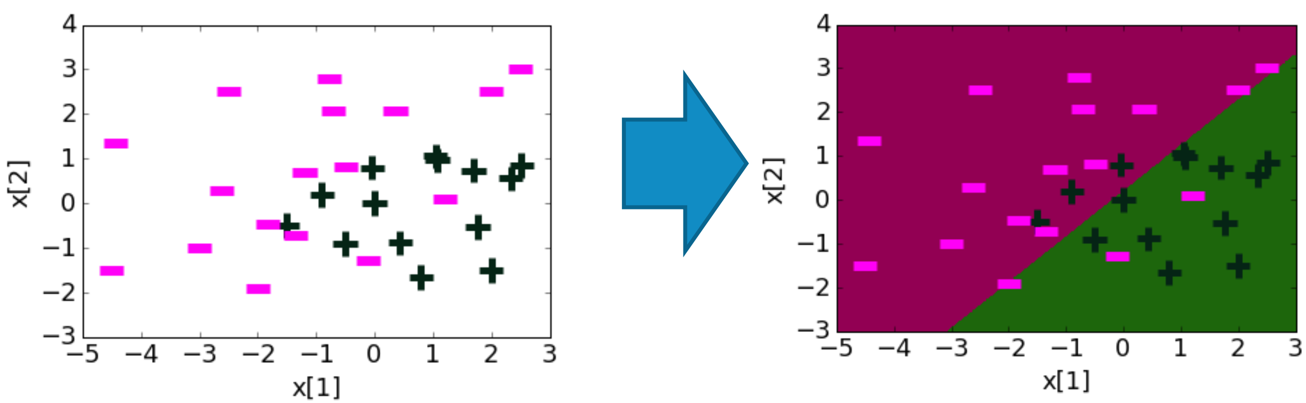
Each split partitions the 2-D space



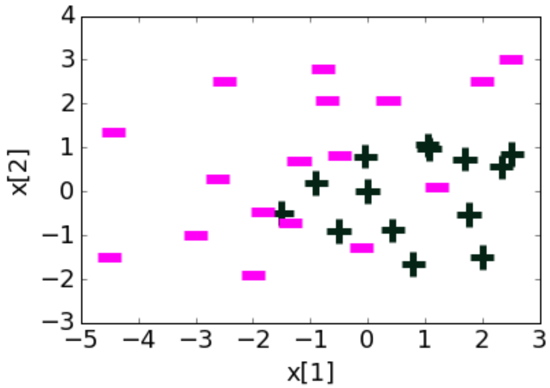
Decision trees vs logistic regression: *Example*

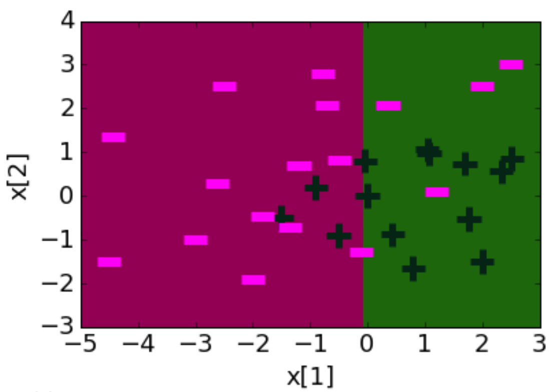
Logistic regression

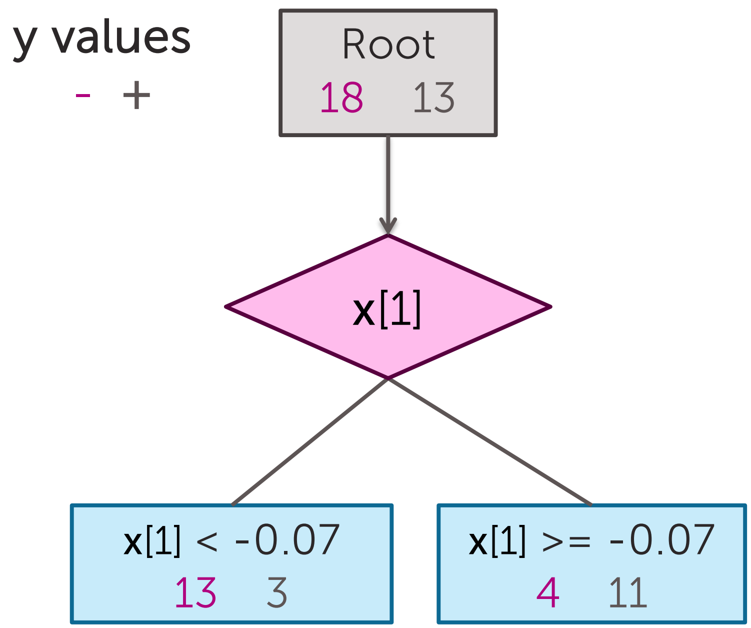
Feature	Value	Weight Learned
$h_0(\mathbf{x})$	1	0.22
$h_1(\mathbf{x})$	x[1]	1.12
$h_2(\mathbf{x})$	x[2]	-1.07



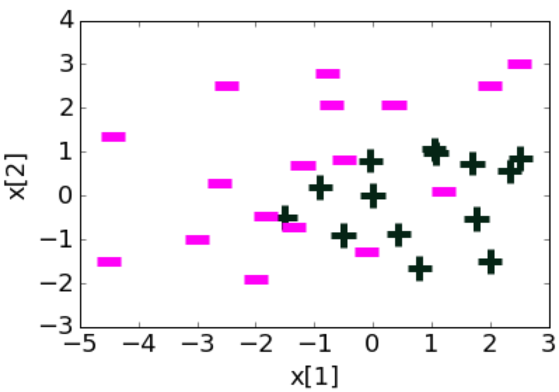
Depth 1: Split on x[1]

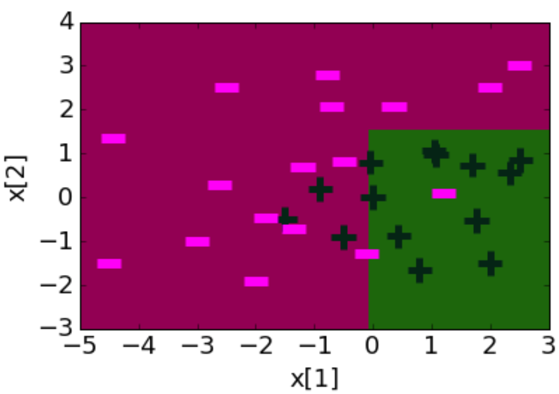


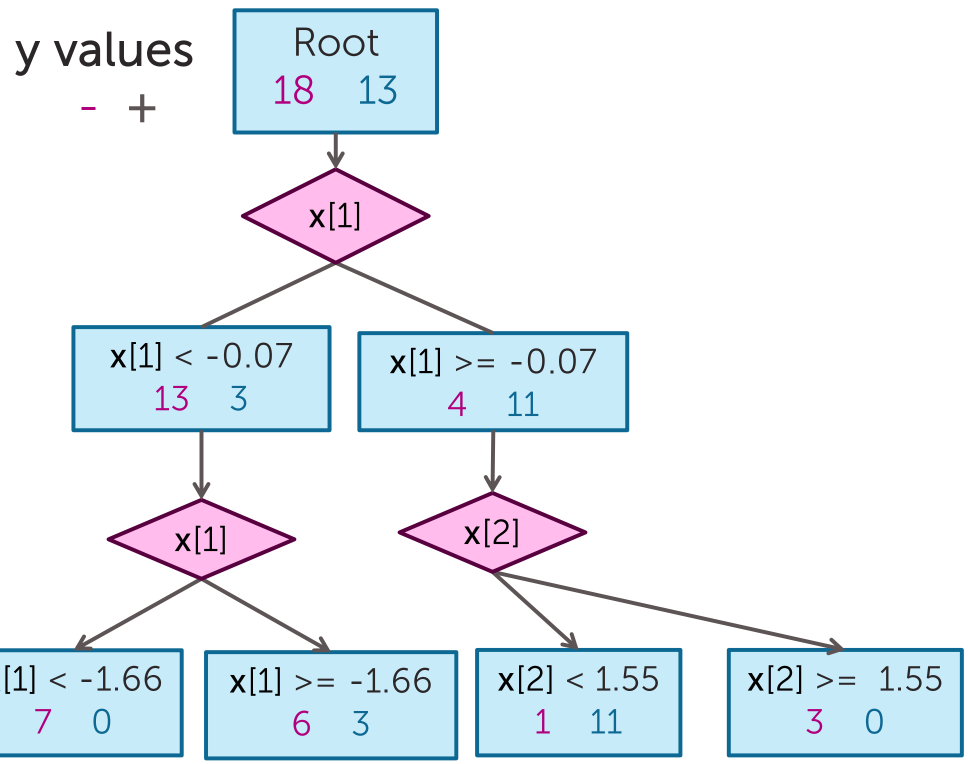




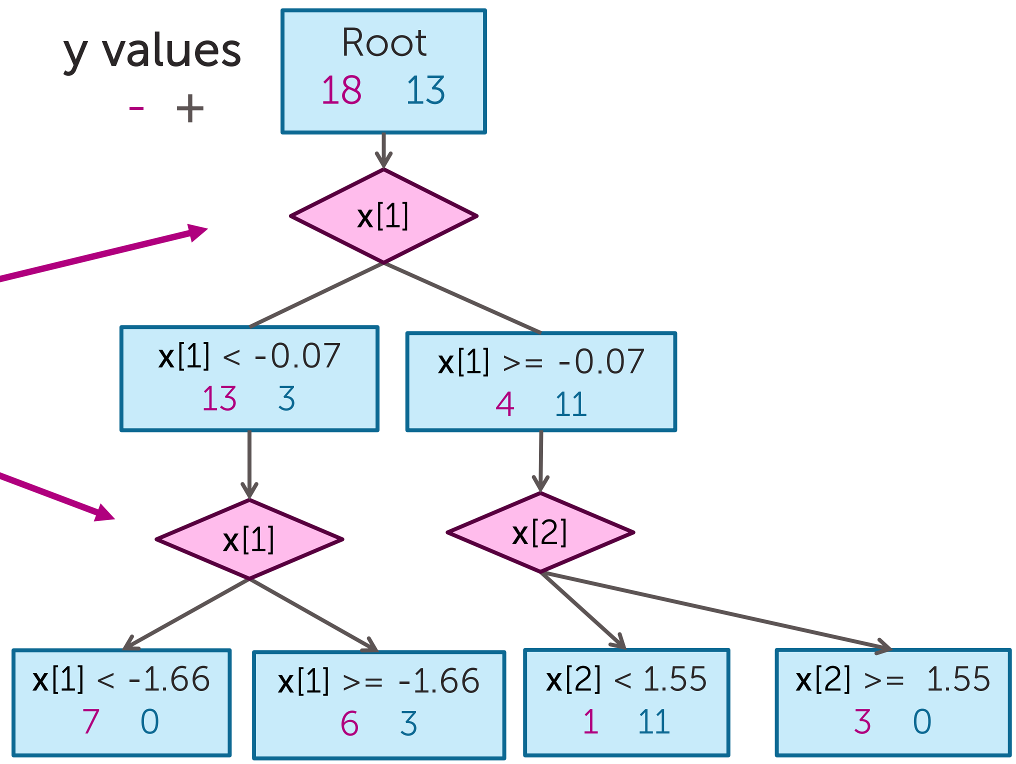
Depth 2


x[2]
x[1]


x[2]
x[1]

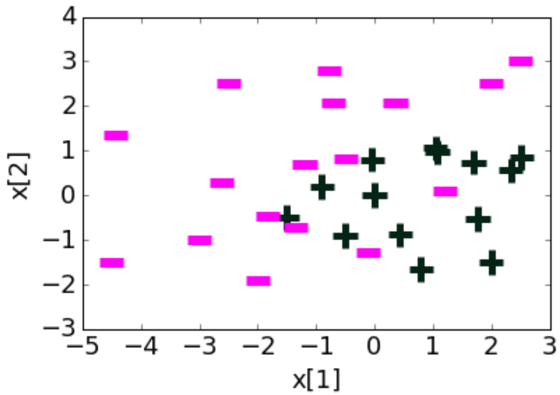

y values
- +
Root
18 13
x[1]
x[1] < -0.07
13 3
x[1] >= -0.07
4 11
x[1]
x[2]
x[1] < -1.66
7 0
x[1] >= -1.66
6 3
x[2] < 1.55
1 11
x[2] >= 1.55
3 0

Threshold split caveat



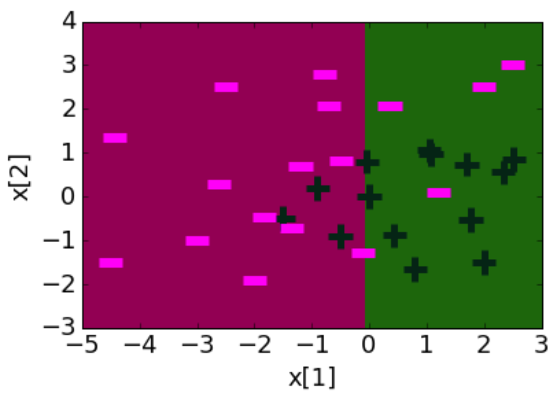
For threshold splits, same feature can be used multiple times

Decision boundaries

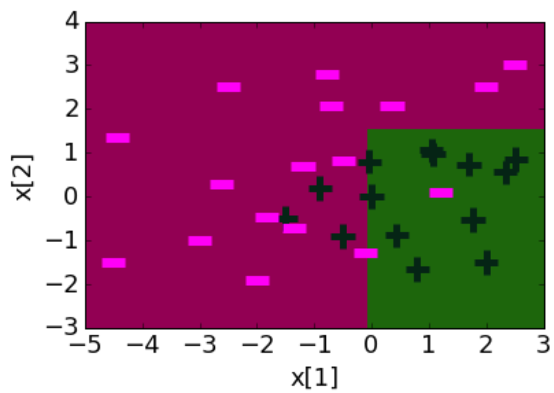


Overfitting ?

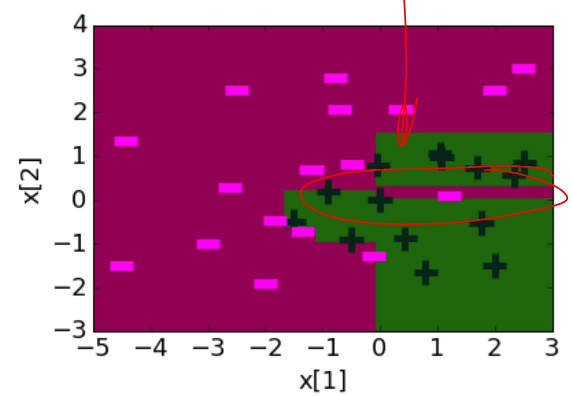
Depth 1



Depth 2

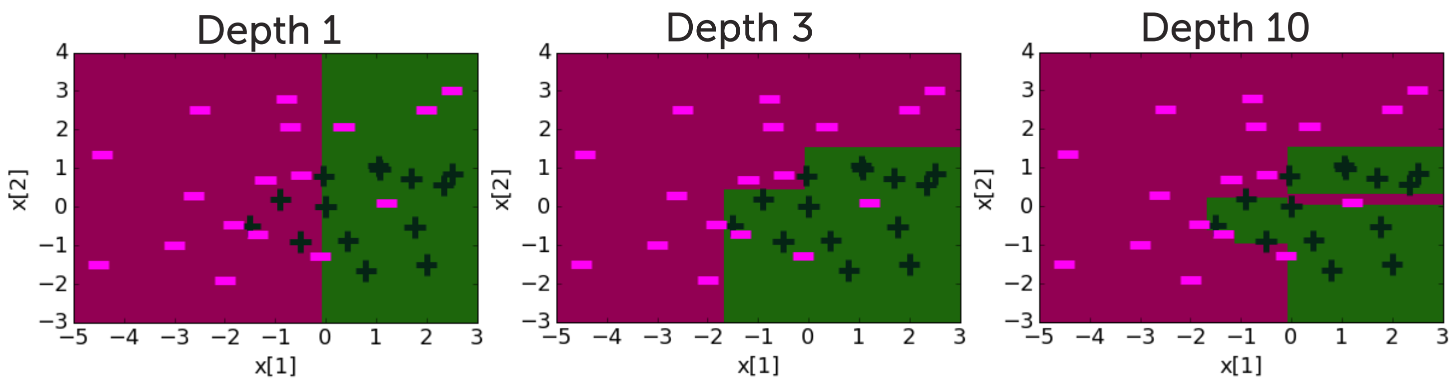


Depth 10

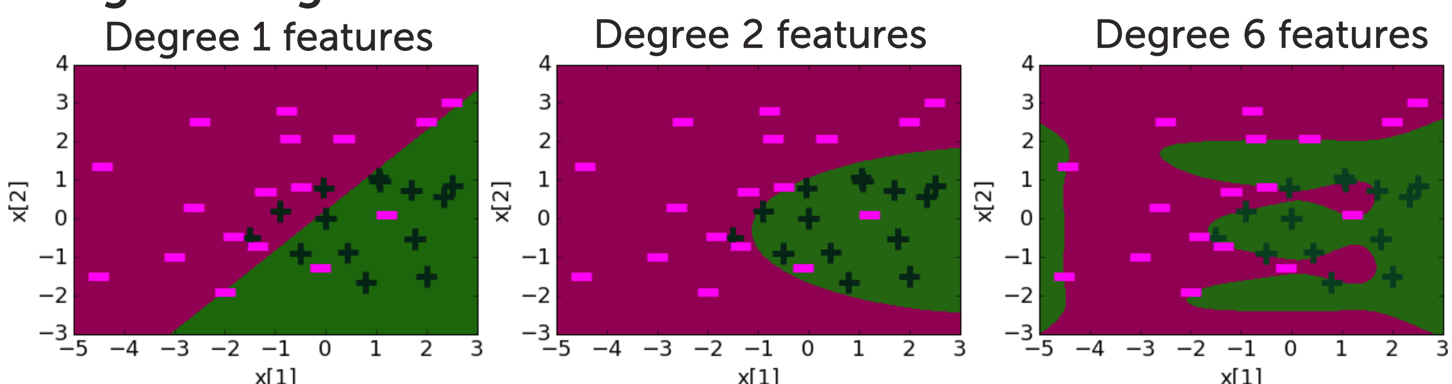


Comparing decision boundaries

Decision Tree



Logistic Regression



Summary of decision trees

What you can do now

- Define a decision tree classifier
- Interpret the output of a decision trees
- Learn a decision tree classifier using greedy algorithm
- Traverse a decision tree to make predictions
  – Majority class predictions
- Tackle continuous and discrete features